[redacted]

EXHIBIT E

12:27:34 IN THE UNITED STATES DISTRICT COURT
FOR THE EASTERN DISTRICT OF TEXAS
MARSHALL DIVISION

IMPLICIT, LLC	)(	
	)(	CIVIL ACTION NO.
VS.	)(	2:18-CV-53-JRG
	)(	MARSHALL, TEXAS DECEMBER 11, 2019
NETSCOUT SYSTEMS, INC.	)(	12:39 P.M.

TRANSCRIPT OF JURY TRIAL

AFTERNOON SESSION

BEFORE THE HONORABLE CHIEF JUDGE RODNEY GILSTRAP,

UNITED STATES DISTRICT JUDGE

APPEARANCES:

FOR THE PLAINTIFF:

SPENCER HOSIE
BRANDON C. MARTIN
FRANCESCA M. S. GERMINARIO
HOSIE RICE, LLP
600 Montgomery Street, 34th Floor
San Francisco, CA 94111

WILLIAM E. DAVIS, III
CHRISTIAN J. HURT
TY WILLIAM WILSON
THE DAVIS FIRM, PC
213 N. Fredonia Street, Suite 230
Longview, Texas 75601

FOR THE DEFENDANT:

ABRAN KEAN
ERIC BURESH
MARK LANG
ERISE, IP
7015 College Boulevard, Suite 700
Overland Park, KS 66211

GIL GILLAM
GILLAM & SMITH, LLP
303 South Washington Avenue
Marshall, TX 75670

COURT REPORTER: Shelly Holmes, CSR, TCRR
Official Court Reporter
United States District Court
Eastern District of Texas
Marshall Division
100 E. Houston
Marshall, Texas 75670
(903) 923-7464

(Proceedings recorded by mechanical stenography, transcript produced on a CAT system.)

12:39:10 PROCEEDINGS

12:39:10 (Jury out.)

12:39:10 COURT SECURITY OFFICER: All rise.

12:39:11 THE COURT: Be seated, please.

12:39:18 Mr. Buresh, are you prepared to continue with your
12:39:31 direct examination?

12:39:32 MR. BURESH: I am, Your Honor.

12:39:32 THE COURT: All right. You may go to the podium.

12:39:35 While he's doing that, Mr. Elliott, please bring
12:39:38 in the jury.

12:39:39 COURT SECURITY OFFICER: All rise.

12:39:40 (Jury in.)

12:40:10 THE COURT: Welcome back from lunch, ladies and
12:40:12 gentlemen. Please have a seat.

12:40:13 When we broke for lunch, the Defendant was
12:40:17 undertaking its direct examination of Mr. Curtin.

12:40:20 And we'll continue with the Defendant's direct
12:40:22 examination of this witness at this time.

12:40:25 Counsel, you may proceed.

12:40:26 MR. BURESH: Thank you, Your Honor.

12:40:26 JOHN CURTIN, DEFENDANT'S WITNESS, PREVIOUSLY SWORN

12:40:26 DIRECT EXAMINATION CONTINUED

12:40:27 BY MR. BURESH:

12:40:27 Q. Mr. Curtin, welcome back.

12:40:31 A. Thank you.

12:40:33 MR. BURESH: Mr. Palisoul, could you please pull
12:40:35 up Slide 14?
12:40:38 Q. (By Mr. Buresh) Are you familiar with flow tables in
12:40:41 the GeoProbe products?
12:40:42 A. Yes, I am.
12:40:44 Q. And can you describe for us the flow table in the
12:40:49 GeoProbe products?
12:40:49 A. The GeoProbe products maintain a flow table in memory,
12:40:53 which tracks and maintains statistics for each flow that we
12:40:59 see in that probe.
12:41:03 Q. So what is the flow table in the GeoProbe products used
12:41:04 for?
12:41:04 A. At the end of a flow, a record is cut and sent to the
12:41:09 next layer from the cabin to the Traffic Processor
12:41:14 containing a summary of that user plane flow.
12:41:18 Q. Does creation of a flow entry create a processing path?
12:41:21 A. No.
12:41:22 Q. What is the relationship between the processing path in
12:41:25 GeoProbes and the flow entry?
12:41:27 A. As packets are going through the processing paths of
12:41:32 the GeoProbe, the flow table is updated with attributes and
12:41:39 counts from those packets.
12:41:40 Q. Is that after processing along the path is already
12:41:43 occurring?
12:41:44 A. Yes.

12:41:44 MR. HURT: Objection, Your Honor. Leading.
12:41:46 THE COURT: Sustained.
12:41:56 Q. (By Mr. Buresh) When along the processing path is the
12:41:59 flow table updated?
12:42:00 A. The flow table is updated at many points during the
12:42:05 processing paths that we've coded.
12:42:07 MR. BURESH: Will you go to Slide 15, please?
12:42:10 Q. (By Mr. Buresh) Do you recognize this as the run time
12:42:20 system we discussed earlier?
12:42:21 A. Yes, I do.
12:42:22 Q. And we're receiving this packet into the system in
12:42:27 operation. Do you see that on the lower right-hand corner?
12:42:31 A. Yes, I do see the packet.
12:42:32 Q. What happens in the GeoProbe products to the packet
12:42:36 when it is received into the system?
12:42:38 A. So in this depiction, the packet is -- is shown
12:42:42 arriving from the EZ Chip into the Cavium. At that point
12:42:48 it is copied into memory on the Cavium. All of the
12:42:51 routines on the Cavium will then use the descriptor to
12:42:55 access that packet whenever they need.
12:42:56 Q. What type of packets are received into the GeoProbe
12:42:59 products?
12:42:59 A. All packets in the GeoProbe products are ethernet
12:43:06 packets.
12:43:06 Q. What is the outermost header on the packet depicted on

12:43:08 Slide 15?
12:43:09 A. The packet on Slide 15 starts with an ethernet header
12:43:14 as the outermost header followed by an IP header, a TCP
12:43:20 header and the control plane data.
12:43:21 Q. How much of this packet is stored in memory?
12:43:25 A. All of this packet is stored in memory on the Cavium.
12:43:27 Q. How long is the packet stored in memory during
12:43:29 processing?
12:43:30 A. So packets are stored in memory for the duration of its
12:43:33 processing through the paths that we've coded on the
12:43:37 Cavium. When it's done going through whatever path it was
12:43:41 assigned, it will be removed from memory and stored on
12:43:45 disks.
12:43:45 Q. Are the headers on this packet stored in memory ever
12:43:57 changed during processing by the GeoProbe?
12:43:58 A. No. We don't change the headers in memory. We need
12:44:02 those headers to access fields in those headers. We need
12:44:07 those headers to get into the payload of the packet, and
12:44:11 ultimately our call trace application needs those headers
12:44:14 to allow network troubleshooting.
12:44:16 Q. Please describe for us the importance of keeping
12:44:19 headers on the packets during processing.
12:44:22 A. So all -- all routines on our Cavium access the packet
12:44:28 through what we described earlier as the descriptor.
12:44:31 That -- the descriptor describes the beginning of the

12:44:34 packet and also offsets into the different layers within
12:44:38 that ethernet packet.
12:44:39 So if the headers were not there, the offsets
12:44:45 would not be valid.
12:44:46 In addition to that, our call trace application
12:44:49 relies on having all headers present.
12:44:51 Q. After processing is complete on a given packet going
12:44:57 through the GeoProbe system, what happens to that packet?
12:44:59 A. When we are done with the -- taking the packet through
12:45:05 the processing paths we've created, it is removed from
12:45:09 memory and stored to disk.
12:45:10 Q. How much of the packet is stored to disk?
12:45:12 A. For control plane packets, all of the packet is stored
12:45:16 to disk. For user plane packets, all of the headers are
12:45:20 stored to disk. And the payload is an optional stored to
12:45:24 disk.
12:45:26 Q. How do the processing routines in the GeoProbe access
12:45:30 information within the packet that's stored in memory?
12:45:34 A. All routines on the Cavium start their processing using
12:45:40 the packet descriptor we described earlier. That packet
12:45:44 descriptor contains an address to the beginning of the
12:45:47 ethernet frame. It also contains offsets to the different
12:45:50 layers within the packet.
12:45:52 Q. Where does the packet descriptor come from?
12:45:55 A. The packet descriptor originated on the EZ Chip, which

12:45:59 did all of our grunt work and initial parsing of every
12:46:02 packet that we see.

12:46:04 MR. BURESH: If you could pull up Slide 16,
12:46:07 please?

12:46:07 Q. (By Mr. Buresh) Mr. Curtin, is this a depiction of the
12:46:17 packet descriptor you've been discussing?

12:46:18 A. Yes, it is.

12:46:19 Q. Is it an accurate depiction?

12:46:21 A. Yes.

12:46:21 Q. How do processing routines that process the packets in
12:46:26 the GeoProbe, how do they use this descriptor to access
12:46:34 information from the packet?

12:46:34 A. So every processing routine on the Cavium which --
12:46:41 start with this descriptor that's shown in the upper left
12:46:45 and the address at the beginning of the ethernet frame,
12:46:49 which is the beginning of the ethernet header on the left
12:46:51 of the packet.

12:46:51 If a routine wanted to access, for example, the
12:46:56 TCP header within this ethernet frame, it would take the
12:47:01 beginning of the ethernet frame address, beginning of the
12:47:04 ethernet header, and add to that value the L4 offset value
12:47:09 from the descriptor. That would allow the routine to begin
12:47:16 accessing the TCP header.

12:47:17 Q. Where are offsets measured from in this packet
12:47:21 descriptor in the GeoProbe?

12:47:23 A. Our offsets are measured from the beginning of the
12:47:27 ethernet frame.
12:47:28 Q. If offsets are measured from the beginning, what would
12:47:33 happen in the GeoProbe if the ethernet header, for example,
12:47:38 was removed?
12:47:40 A. If the ethernet header was removed, then any routine
12:47:46 trying to access a subsequent layer of that packet using
12:47:48 the offsets we've created would fail to find the layer it
12:47:52 was looking for; would likely crash the software.
12:47:55 Q. How would routines be able to find the information they
12:47:59 were looking for if a header had been removed?
12:48:01 A. As written in our code, they would not be able to find
12:48:07 anything.
12:48:08 Q. How important is it in the GeoProbe to keep the headers
12:48:11 on the packet at all times?
12:48:13 A. We need to keep the headers of all packets at all times
12:48:18 in order for our routines to be able to access those
12:48:21 headers, as needed, during our processing paths. It's also
12:48:25 vital for our call trace application to be able to rebuild
12:48:29 the subscriber activity from all layers of the network for
12:48:34 troubleshooting.
12:48:35 Q. Is this packet descriptor passed to the processing
12:48:38 routines on the path -- processing path?
12:48:40 A. Yes, it is.
12:48:45 Q. Could you show us in the source code an example of

12:48:45 where that happens?

12:48:45 A. Yes, I can.

12:48:47 MR. BURESH: Mr. Palisoul, can you do the split
12:48:49 screen and pull up DX-454 at 745 and 747?

12:49:03 Q. (By Mr. Buresh) What are we seeing in this source
12:49:04 code, Mr. Curtin?

12:49:05 A. This source code is the structure definition of our
12:49:09 packet descriptor, and the zoomed-in portions show the
12:49:14 offset values from that packet descriptor. Highlighted are
12:49:17 the L3Offset, the L4Offset and the appDataOffset.

12:49:27 Q. Can you show me in the source code where this packet
12:49:29 descriptor would be provided to one of the processing
12:49:31 routines?

12:49:31 A. Yes, I can.

12:49:33 MR. BURESH: Mr. Palisoul, can you pull up in this
12:49:37 same split screen format the DX-453 at 523.

12:49:45 Q. (By Mr. Buresh) And Mr. Curtin, can you describe for
12:49:47 us what we're seeing in this source code?

12:49:53 A. Yes. This shows the routine definition of the TCP
12:49:59 state machine processing routine, and that's highlighted in
12:50:01 yellow. And highlighted in green is a pointer to the
12:50:05 packet descriptor as an input to that routine.

12:50:08 Q. So which routine is receiving the packet descriptor
12:50:17 here?

12:50:17 A. So the TcpSm_processPacket routine is receiving a

12:50:27 pointer to the packet descriptor as an input.
12:50:27 Q. Would each routine in a processing path similarly
12:50:30 receive the packet descriptor?
12:50:31 A. Yes.
12:50:33 Q. How does each routine use the packet descriptor to
12:50:37 access information within this example packet we're looking
12:50:41 at?
12:50:41 A. So in this example, the TcpSm_processPacket routine
12:50:53 would use the packet descriptor to find the beginning of
12:50:56 the frame in memory, and then it would use the L4Offset
12:51:02 from the descriptor to then access the TCP header within
12:51:05 that ethernet frame.
12:51:06 Q. How does the use of offsets impact the headers on the
12:51:09 packet and memory?
12:51:11 A. The -- we don't change the headers by use of the offset
12:51:18 values.
12:51:20 Q. What is the outermost header of packets stored in
12:51:26 memory in the GeoProbe products?
12:51:28 A. Ethernet.
12:51:29 Q. How do you know that?
12:51:31 A. We only accept ethernet packets into our products, and
12:51:36 we don't change any of them once they come in. So the --
12:51:39 the outermost header of all of the packets has to be
12:51:42 ethernet.
12:51:43 Q. Do the GeoProbe products convert packets from one

12:52:00 format to another?
12:52:01 A. No.
12:52:03 Q. I'd like to show you a few slides now, and I know you
12:52:11 haven't seen these before.
12:52:12 MR. BURESH: But if we could bring up Slide 118
12:52:15 from Dr. Almeroth's testimony. And just focus on the
12:52:25 source code on the right-hand side, please.
12:52:26 Q. (By Mr. Buresh) Do you recognize this source code,
12:52:32 Mr. Curtin?
12:52:34 A. Yes, I do.
12:52:35 Q. And what does this code do?
12:52:37 A. This is the TCP state machine process packet routine.
12:52:45 It received as input arguments, among other things, the
12:52:50 pointer to the packet descriptor, and then in the
12:52:53 highlighted red portion at the bottom, it is -- is creating
12:53:00 a pointer to the TCP header within the ethernet packet
12:53:05 represented by that packet descriptor.
12:53:08 Q. What impact does creating a pointer to a TCP header
12:53:13 have on the packet that's stored in memory?
12:53:18 A. A pointer to a layer within the ethernet packet does
12:53:23 not affect the headers of the packet in memory.
12:53:33 MR. BURESH: If you go to Slide 119.
12:53:35 Q. (By Mr. Buresh) And, again, do you recognize the lines
12:53:46 of code at the upper portion of this screen?
12:53:51 A. I do. There's not much context around them.

12:53:56 Q. Do you recognize the routines that would be indicated
12:54:00 here?
12:54:00 A. I do know the set application ID routine.
12:54:06 Q. And what does that do?
12:54:08 A. It sets the application ID field within the packet
12:54:12 descriptor.
12:54:13 Q. What impact does that processing routine have on the
12:54:21 packets stored in memory?
12:54:23 A. The packets stored in memory is not changed by this
12:54:29 routine.
12:54:41 MR. BURESH: If we can go to Slide 120. Slide
12:54:59 120. Thank you.
12:55:00 Q. (By Mr. Buresh) And, again, we have some code on the
12:55:02 center portion of the screen here that's being blown up to
12:55:07 where it was before.
12:55:09 MR. BURESH: My apologies.
12:55:10 Q. (By Mr. Buresh) Do you recognize the code on this
12:55:13 screen, Mr. Curtin?
12:55:14 A. Yes, I do.
12:55:17 Q. And what is this code doing?
12:55:26 A. Well, the highlighted portion is just declaring a
12:55:30 pointer to the application layer of the ethernet frame.
12:55:37 Q. Does this code convert a packet?
12:55:39 A. No.
12:55:40 Q. What impact does this code have on the packet that's

12:55:42 being processed in the GeoProbe?
12:55:45 A. No impact on the packet in memory.
12:55:53 MR. BURESH: If we could go back to our slides
12:55:55 now.
12:55:57 Q. (By Mr. Buresh) Are you familiar with the concept of
12:56:02 reassembly?
12:56:03 A. Yes.
12:56:03 Q. In the GeoProbe products?
12:56:05 A. Yes.
12:56:05 Q. And could you describe for the jury how reassembly
12:56:23 operates in the GeoProbe products?
12:56:25 A. Yes. So in this example, we have three ethernet
12:56:31 packets that were received into the GeoProbe. Each has the
12:56:35 same header structure, the headers being ethernet, IP, and
12:56:40 TCP, and each carrying SIP payload data.
12:56:47 As part of our reassembly, we can create a new
12:56:51 synthetic ethernet packet that joins those SIP data pieces
12:56:56 into a single SIP message. We do that by first copying the
12:57:01 headers and the payload of the first packet in the
12:57:05 sequence.
12:57:05 In this example, it's the third packet received on
12:57:07 the far right. A complete copy is made of that packet.
12:57:15 Then on the second packet in the sequence, since
12:57:17 the headers are the same, the header structures are the
12:57:20 same, we will take the payload data and add that to the end

12:57:23 of the synthetic ethernet packet we were -- we had made a
12:57:28 copy of.

12:57:29 And then the same happens on the third packet in
12:57:32 the sequence. The header structures are the same. We will
12:57:36 take the SIP payload data and add it on to the end of our
12:57:43 new synthetic ethernet packet.

12:57:48 Q. What's happening to the packets that were received in
12:57:50 the GeoProbe?

12:57:50 A. So the three packets at the bottom are not changed in
12:57:54 memory.

12:57:54 Q. What's the relationship between the headers on the
12:57:57 three packets that were received?

12:58:00 A. All three packets would have the same header structure,
12:58:04 being ethernet, IP, and TCP, in this case.

12:58:09 Q. Does the synthetic packet have the same header
12:58:12 structure as the other three?

12:58:14 A. Yes, it does have the same header structure. The
12:58:16 headers are a copy of the headers from the first packet in
12:58:19 the sequence.

12:58:19 Q. What is the outermost header on each of the packets
12:58:22 that are received by the GeoProbe in this example?

12:58:24 A. The outermost header on all three of the packets
12:58:29 received is the ethernet header.

12:58:31 Q. What is the outermost header on the synthetic packet
12:58:35 that's created as a result of this reassembly?

12:58:38 A. The outermost header on the synthetic packet is also an
12:58:45 ethernet header.
12:58:45 Q. Does this process result in a conversion of any packets
12:58:46 at all?
12:58:47 A. No.
12:59:00 MR. BURESH: Your Honor, I pass the witness.
12:59:01 THE COURT: All right. Cross-examination by the
12:59:02 Plaintiff.
12:59:16 MR. HURT: Your Honor, may I approach the witness?
12:59:20 THE COURT: You may.
12:59:34 All right. Mr. Hurt, you may proceed with
12:59:37 cross-examination.
12:59:37 CROSS-EXAMINATION
12:59:38 BY MR. HURT:
12:59:38 Q. Good afternoon, Mr. Curtin.
12:59:42 A. Good afternoon.
12:59:43 Q. You're not an expert witness for -- for NetScout; is
12:59:46 that right?
12:59:46 A. No.
12:59:47 Q. And so you're not telling the jury one way or the other
12:59:52 about whether NetScout infringes Implicit's patents, are
12:59:55 you?
12:59:55 A. No. I'm just here to explain our product.
12:59:58 Q. And you didn't take the claim terms in the patent and
01:00:04 compare them up to NetScout's product to determine any

01:00:07 issue of infringement, did you?
01:00:08 A. No.
01:00:09 Q. Okay. And whether NetScout infringes, that's an issue
01:00:14 ultimately for the experts in this case, isn't it?
01:00:16 A. Yes.
01:00:18 Q. And NetScout has its expert, Dr. Jeffay, right?
01:00:21 A. I -- I believe that's the case.
01:00:24 Q. And you mentioned on direct that the GeoProbe products
01:00:29 are all written in the C programming language; is that
01:00:32 correct?
01:00:32 A. That's correct.
01:00:34 Q. And that's a standard programming language that -- that
01:00:38 a lot of computer programmers know, right?
01:00:41 A. Yes.
01:00:41 Q. It's not something you need to work at NetScout, to
01:00:44 understand how C code operates, right?
01:00:47 A. No. C is not a proprietary language.
01:00:51 Q. And that's true for how pointers work?
01:00:54 A. Yes.
01:00:54 Q. And data structures?
01:00:56 A. Yes.
01:00:56 Q. And functions and routines, right?
01:00:58 A. Yes.
01:01:20 MR. HURT: Mr. Diaz, if you could pull up
01:01:23 Slide 10.2 of Mr. Curtin's direct demonstratives.

01:01:27 Q. (By Mr. Hurt) Do you remember, Mr. Curtin, discussing
01:01:31 this series of -- of pipes during your direct examination?

01:01:35 A. Yes, I do.

01:01:35 Q. And in this situation on the screen, is it showing
01:01:41 those pipes for a control protocol packet?

01:01:48 A. It -- it does show that processing path, yes.

01:01:50 Q. And just so I understand this, this series of slides,
01:01:54 this isn't an internal engineering presentation that y'all
01:01:58 had at NetScout, is it?

01:01:59 A. No, it is not.

01:02:01 Q. It's something you guys did for this case?

01:02:04 A. Yes.

01:02:04 Q. And its intent was to teach the jury and give, I guess,
01:02:10 a simplified version of how the GeoProbe works; is that
01:02:10 fair?

01:02:13 A. It is a bit simplified.

01:02:15 Q. And the control protocols over HTTP module, one of
01:02:21 those control product -- protocols would be something like
01:02:26 SIP, right?

01:02:26 A. SIP would usually fall in the normal control protocols
01:02:33 module. There is one case, I think, one customer that does
01:02:36 do SIP over HTTP.

01:02:38 Q. Well, what about something like XCAP over HTTP?

01:02:42 A. XCAP is always over HTTP.

01:02:45 Q. Or -- and then another example of a Layer 7 control

01:02:50 protocol you mentioned was Diameter, right?
01:02:51 A. Yes.
01:02:52 Q. And those protocols, SIP, Diameter -- I believe the
01:02:56 third one, you mentioned GTPC, those are all Layer 7
01:03:00 protocols, true?
01:03:00 A. Yes.
01:03:00 Q. And being able to identify and process those protocols
01:03:04 is why companies like AT&T buy y'all's products, right?
01:03:07 A. One of the reasons.
01:03:10 Q. In fact, isn't -- if you were unable to identify those
01:03:14 protocols, isn't it true, sir, that they wouldn't buy those
01:03:17 products?
01:03:18 A. If we could not identify control plane protocols, then
01:03:22 we would not able to -- yeah, we wouldn't be able to
01:03:26 process them and do any of our metrics.
01:03:28 Q. And the metrics you mentioned -- you mentioned some
01:03:30 statistics and something called call trace. Do you
01:03:34 remember that?
01:03:35 A. Yes.
01:03:35 Q. And all of your customers use call trace, right?
01:03:38 A. Yes, they do.
01:03:39 Q. And that includes all the service providers; AT&T,
01:03:42 Verizon, T-Mobile?
01:03:43 A. That's correct.
01:03:44 Q. And to be able to see those statistics in call trace,

01:03:49 you've got to be able to identify things like SIP,
01:03:53 Diameter, GTPC, don't you?
01:03:56 A. That's correct.
01:03:57 Q. And you mentioned SIP may or may not be over HTTP; is
01:04:02 that right?
01:04:02 A. That's correct.
01:04:03 Q. And it's one of those Layer 7 control protocols?
01:04:06 A. SIP is a control protocol, yes.
01:04:08 Q. And here you have one pipe for a control protocol -- I
01:04:15 guess I'll just call it a pipe; is that right?
01:04:17 A. Could you --
01:04:18 Q. Sure. In this diagram, there's just one pipe for all
01:04:21 the control protocols over HTTP, right?
01:04:23 A. It's depicted as a simplification. If there needs to
01:04:31 be specific paths within that pipe for SIP versus Diameter,
01:04:35 we, of course, have different paths within there.
01:04:38 Q. Okay.
01:04:40 MR. HURT: And, Mr. Diaz, if you could pull up
01:04:44 Defendant -- or, I'm sorry, PTX Exhibit 5, and the page is
01:04:53 41.
01:04:55 Q. (By Mr. Hurt) And this document, Mr. Curtin, is from
01:05:00 the internal GeoProbe engineering document, right?
01:05:04 A. Yes, it is.
01:05:05 Q. And it shows an overview of how the GeoProbe works,
01:05:09 doesn't it?

01:05:11 A. This slide describes the SIP protocol and how we
01:05:16 identify and distribute SIP packets.
01:05:20 Q. And for SIP, on this slide alone, there's four flavors
01:05:24 of SIP, right?
01:05:26 A. Yes.
01:05:26 Q. There's SIP, SIP/IPsec, SIP/TLS, and WebSocket. Do you
01:05:38 see those?
01:05:38 A. I do see those.
01:05:38 Q. That's not just one pipe per SIP, right?
01:05:38 A. No.
01:05:38 Q. And then each of those application protocols run on
01:05:42 something called a stack. You see that on the next column?
01:05:45 A. Yes.
01:05:45 Q. And for SIP by itself, it can go on UDP, TCP, and SCTP.
01:05:53 Do you see that?
01:05:54 A. Yes, I do.
01:05:54 Q. And SIP/IPsec can go over four different stacks, right?
01:05:59 A. That's how it's shown.
01:06:00 Q. And SIP/TLS is the TLS/TCP stack, right?
01:06:13 A. That's correct.
01:06:14 Q. And WebSocket is SIP/HTTP/TLS/TCP, true?
01:06:14 A. That's true.
01:06:14 Q. And those three on top, SIP, HTTP, and TLS, those are
01:06:19 all in the application layer, aren't they?
01:06:21 A. They're all in the TCP payload, so from a TCP

01:06:27 perspective, they're all in the application payload of TCP.
01:06:30 Q. So we've got four SIP protocols, and we've got six
01:06:37 different stacks. And then in addition to that, we have to
01:06:41 do reassembly, which is two columns over. Do you see that?
01:06:45 A. I do see the reassembly column.
01:06:48 Q. And so for each of these protocols, we've got to do
01:06:51 something called IP defrag? Do you see that?
01:06:55 A. Yes.
01:06:55 Q. And TCP reassembly? Do you see that?
01:06:58 A. Yes, I do.
01:06:59 Q. And in addition for the SIP/TLS, does that refer to
01:07:04 encrypted traffic, sir?
01:07:06 A. Yes, it does.
01:07:06 Q. And in that case, we've got to do TCP reassembly on
01:07:11 that encrypted traffic, right?
01:07:12 A. We have to do TCP reassembly for all of the SIP cases
01:07:16 over TCP.
01:07:17 Q. So when you showed one pipe for those control
01:07:21 protocols, it's really not one pipe, right?
01:07:23 A. It's a simplification to show going down a pipe or a
01:07:29 path for a control plane. Once processing those control
01:07:33 plane paths, there are more and more paths for each of the
01:07:36 specific control plane protocols.
01:07:38 Q. But that one -- when you showed it as one pipe for
01:07:42 these control plane protocols, it is not one pipe; is that

01:07:47 fair?
01:07:47 A. I don't believe I agree with that statement. There is
01:07:52 a point in the code where there's a path taking down a -- a
01:07:56 control plane path that further goes into other paths for
01:08:02 each of the different control plane protocols.
01:08:07 MR. HURT: I'm going to object. Non-responsive,
01:08:09 Your Honor.
01:08:09 THE COURT: The portion of the answer that says,
01:08:27 "I don't believe I agree with that statement," is
01:08:28 responsive. The remainder is non-responsive. So I will
01:08:32 strike the remainder of the witness's answer after he said,
01:08:35 "I don't believe I agree with that statement."
01:08:37 Let's proceed.
01:08:39 Q. (By Mr. Hurt) So for SIP, there's four different types
01:08:42 of protocols shown here just for SIP, right?
01:08:45 A. I didn't quite understand that question.
01:08:46 Q. For the SIP protocol, the internal slides y'all have
01:08:49 shows there's four different varieties of SIP; is that
01:08:54 correct?
01:08:54 A. Okay. I understand your question. There are four
01:09:00 different ways the SIP can be carried, yes.
01:09:00 Q. And that's not -- that's not just a single pipe, as you
01:09:04 showed it in your demonstrative, true?
01:09:06 A. There is not a single processing path for all SIP
01:09:12 traffic.

01:09:13 Q. And, in fact, you don't know which one of these four
01:09:19 SIP protocols it is until the GeoProbe actually looks at
01:09:23 the packet, right?
01:09:28 A. Could you repeat that question?
01:09:31 Q. Sure.
01:09:31 A. Yeah.
01:09:31 Q. So if there's a packet that comes into the GeoProbe,
01:09:34 GeoProbe doesn't know if it's SIP or SIP/IPsec or SIP/TLS
01:09:41 or WebSocket until you've actually looked at it, right?
01:09:44 A. Well, right. The packet will come into our probe, and
01:09:47 we will have to parse that packet to understand it.
01:09:49 Q. And you've got to figure out is it getting carried on
01:09:54 UDP, TCP, or SCTP as part of that, don't you?
01:09:58 A. Correct. Part of our EZ Chip processing identifies the
01:10:03 layers of the packet.
01:10:04 Q. And you don't do that until you've actually received
01:10:06 the packet, right?
01:10:07 A. We can't parse a packet we haven't received.
01:10:11 Q. And in addition to the EZ Chip processing, you've got
01:10:14 to do TCP reassembly to figure out if it's SIP, true?
01:10:19 A. No. We'll know it's SIP through either the Layer 4
01:10:27 port or from a tech scan we have on the EZ Chip.
01:10:31 Q. But reassembly is done for SIP; is that right?
01:10:35 A. We do reassembly for SIP.
01:10:36 Q. And you don't know if you're going to do reassembly or

01:10:39 not until the packet has already got there, true?
01:10:42 A. The EZ Chip will identify the packet as a SIP packet
01:10:46 based on either the L4 port or a tech scan in the EZ Chip.
01:10:51 From that point, we know we will eventually do reassembly
01:10:58 on that packet.
01:10:58 Q. Before the packet comes to the GeoProbe, the GeoProbe
01:11:00 does not know if it's going to have to do reassembly or not
01:11:03 on that packet; is that true?
01:11:06 A. We don't know what we're going to do with a packet we
01:11:11 haven't received.
01:11:12 THE COURT: Counsel, approach the bench, please.
01:11:15 (Bench conference.)
01:11:24 THE COURT: Mr. Buresh, during this
01:11:27 cross-examination, you may not be aware of it, but you are
01:11:31 making a lot of facial expressions. You are rolling your
01:11:35 eyes, your eyebrows, you're dipping your head. I assume
01:11:39 you're not intentionally trying to communicate with the
01:11:41 witness or the jury, but you need to sit there and be still
01:11:45 and not engage in any non-verbal communications --
01:11:50 MR. BURESH: Yes, Your Honor.
01:11:51 THE COURT: -- intentional or otherwise.
01:11:51 MR. BURESH: I didn't --
01:11:55 THE COURT: It's time to -- it's time to put your
01:11:56 poker face on.
01:11:57 MR. BURESH: Thank you, Your Honor.

01:11:58 THE COURT: Let's continue.

01:11:59 (Bench conference concluded.)

01:12:09 THE COURT: Let's proceed.

01:12:14 MR. HURT: Mr. Diaz, can you go to the GeoProbe
01:12:17 flow table demonstrative from Mr. Curtin's slides?

01:12:26 Q. (By Mr. Hurt) Do you recall, Mr. Curtin, describing
01:12:34 the GeoProbe flow table on your -- during your direct
01:12:37 testimony?

01:12:37 A. Yes, I do.

01:12:38 Q. And you mentioned that it contains a number of
01:12:40 statistics about the flow, right?

01:12:41 A. Yes.

01:12:43 Q. But in addition to that, it also identifies the
01:12:46 protocol as shown here in the protocol ID, right?

01:12:48 A. That's correct.

01:12:49 Q. And it also shows what application you have, true?

01:12:53 A. Yes.

01:12:55 Q. And isn't it correct that the GeoProbe uses that
01:12:58 information as part of processing the packets?

01:13:01 A. The protocol ID does determine which processing path we
01:13:06 take through our system, yes.

01:13:07 Q. And so that's just not some statistic y'all track?

01:13:10 A. That's correct.

01:13:11 Q. Okay. And for these flow entries, there's a key, and
01:13:16 that's referring to the tuple key, right?

01:13:19 A. The key to the flow table is a 5-tuple, plus a couple
01:13:25 of other items.
01:13:27 Q. And you get that from the first packet that comes into
01:13:31 the GeoProbe, don't you?
01:13:32 A. We look up the key for every packet that comes in.
01:13:34 Q. Sure. Let me ask it this way: The first packet of a
01:13:37 flow has yet to arrive to the GeoProbe. Is there an entry
01:13:41 for that flow in the flow table?
01:13:42 A. If we have not received that key value, then, no, there
01:13:46 would be nothing in that flow table for that key table --
01:13:49 Q. So then the packet comes in, and from that packet you
01:13:53 obtain the key value, right?
01:13:55 A. For each packet that comes in, we will extract the
01:14:01 fields from that packet that make up the key.
01:14:05 Q. That's TCP source port, TCP destination port, IP
01:14:08 address source, IP destination, right?
01:14:09 A. That's not entirely correct. Would you mind if I
01:14:11 corrected it specifically?
01:14:13 Q. Well, are those four included?
01:14:14 A. Those four are included if it's a TCP packet within the
01:14:19 ethernet frame.
01:14:19 Q. Understood. And at that point in time, after -- after
01:14:24 that first packet is received and the tuple is -- is
01:14:30 extracted, the GeoProbe then creates a flow entry, doesn't
01:14:34 it?

01:14:34 A. That's correct.

01:14:35 Q. And as it processes that flow, the GeoProbe fills in 01:14:40 the protocol ID, right?

01:14:43 A. The -- every packet will have a protocol ID from the 01:14:48 EZ Chip, and then when we create the entry in the flow 01:14:53 table, we will populate that protocol based on what the 01:14:57 EZ Chip gave us.

01:14:58 Q. And with the application ID, once the GeoProbe finds 01:15:02 out the application ID, it then populates the flow table 01:15:05 entry with that value, right?

01:15:07 A. That's correct.

01:15:08 Q. Okay. And that information isn't in the flow table for 01:15:13 the first packet when the flow arrives, true?

01:15:17 A. Yes, that's true, if we did not receive the packet, 01:15:23 there'd be nothing in the flow table.

01:15:44 MR. HURT: Ms. Lockhart, may I have the ELMO?

01:16:07 Q. (By Mr. Hurt) Do you recall, Mr. Curtin, on direct 01:16:09 being asked a series of questions about some case 01:16:12 statements from a function called 01:16:17 imon_Reassembly_processPacket?

01:16:18 A. I don't remember seeing which function they were in.

01:16:22 Q. Okay. But do you remember seeing case statements that 01:16:24 looked like these on the screen?

01:16:25 A. Yes.

01:16:28 Q. And these case statements tell you as you're going

01:16:32 along which path to take, right?
01:16:34 A. Yes, the protocol ID indicates which of the paths we
01:16:49 will go down.
01:16:49 Q. So if you -- if you have, let's say for case SIP, you
01:16:53 see that here on Line 2763?
01:16:57 A. Okay.
01:16:58 Q. You got to this point -- and let me ask this first.
01:17:00 Fsb.c, FSB stands for flow state block, right?
01:17:04 A. FSB stands for flow state block.
01:17:08 Q. And that's the series of code that's responsible for
01:17:12 the flow state processing that uses the flows table, right?
01:17:15 A. Yes.
01:17:15 Q. Because the flow table is called the flow state block
01:17:19 table, true?
01:17:20 A. Yes.
01:17:20 Q. So as the packet is being processed, at this point in
01:17:23 time, it may not have determined if it's SIP, right?
01:17:27 A. No, we would know the packet was SIP before this point.
01:17:30 Q. And then at this point, if it has determined it's SIP,
01:17:35 the code then does something, right?
01:17:36 A. At this point, based on the fact that we know this
01:17:40 packet is SIP, there are instructions to follow the next
01:17:45 path.
01:17:46 Q. But you don't actually go down that path until you hit
01:17:52 this case statement, right?

01:17:53 A. Well, as part of writing our code, we order all of our
01:18:00 routines, or case statements, or "if" statements. And so
01:18:05 we determine when we go down this path of code.
01:18:09 Q. When we're at this point in code -- in the code and
01:18:12 there's a series of case statements, those are all
01:18:16 potential options; is that fair?
01:18:21 A. Those are all potential paths within our code, yes.
01:18:24 Q. Okay. And then at this point in the code is where you
01:19:05 figure out which one of those potential paths you're going
01:19:08 to go down; is that fair?
01:19:11 A. We've written the code to define all of these paths.
01:19:14 So as the packet comes in and goes through this code, it's
01:19:18 going to take the path we designed for it.
01:19:21 MR. HURT: I'm going to object. Non-responsive,
01:19:24 Your Honor.
01:19:24 THE COURT: After the answer, okay, the remainder
01:19:36 of that answer I find is non-responsive, and I'll strike
01:19:40 it.
01:19:41 Mr. Curtin, you need to limit your answers to the
01:19:41 questions asked. Mr. Buresh is going to get a chance to
01:19:41 follow up and ask you questions he thinks need to be asked
01:19:48 after this cross-examination.
01:19:48 THE WITNESS: All right. Yeah.
01:19:50 THE COURT: Just a minute. All right?
01:19:56 THE WITNESS: Yes, sir.

01:19:57 THE COURT: "Yeah" is not really a great response
01:20:01 to a United States District Court.

01:20:02 THE WITNESS: Yes, sir.

01:20:03 THE COURT: Thank you. Let's continue.

01:20:04 MR. HURT: Thank you, Your Honor.

01:20:05 Q. (By Mr. Hurt) At this point in the code, Mr. Curtin,
01:20:08 the GeoProbe is picking one of those possible paths that it
01:20:11 will actually process the packet on; is that fair?

01:20:20 A. Yes.

01:20:22 Q. Okay.

01:20:29 MR. HURT: I want to go, if you can, Mr. Diaz, to
01:20:32 the last slide in Mr. Curtin's direct, the one that
01:20:41 depicts -- thank you.

01:20:43 Q. (By Mr. Hurt) Do you remember discussing this slide in
01:20:46 your direct testimony, Mr. Curtin?

01:20:49 A. Yes, I do.

01:20:50 Q. And this is showing how TCP reassembly for control
01:20:53 plane packets operates in GeoProbe, right?

01:20:56 A. Yes.

01:20:57 Q. And we have this first received packet on the bottom
01:21:00 left that says TCP sequence No. 3, right?

01:21:05 A. That's correct.

01:21:05 Q. And in this demonstrative, you have sequence No. 3
01:21:11 means we got this one first, but in the ultimate message,
01:21:15 it should have been third; is that right?

01:21:17 A. That's correct.

01:21:18 Q. And as part of this process -- well, let me ask, this
01:21:22 first received packet has something called SIP data in it.
01:21:27 Do you see that?

01:21:28 A. Yes.

01:21:29 Q. And that's the TCP segment, right?

01:21:36 A. Oh, yes, the SIP data makes up the TCP segment.

01:21:42 Q. Otherwise known as the payload?

01:21:44 A. Yes, the payload of the TCP packet, that's correct.

01:21:47 Q. And that -- that payload of the TCP packet when it's
01:21:50 SIP, it could have a SIP protocol header on it, true?

01:21:54 A. Could you state that one again?

01:22:01 Q. Sure. So SIP is a protocol, isn't it?

01:22:04 A. Yes, it is.

01:22:05 Q. It's an application layer protocol; is that true?

01:22:10 A. Yes, that's true.

01:22:11 Q. And part of the SIP standard is that there is something
01:22:14 called a SIP header, correct?

01:22:15 A. That's correct.

01:22:16 Q. And the SIP header is going to be reflected inside the
01:22:21 SIP data portion shown on your demonstrative, right?

01:22:23 A. Yes, that's correct.

01:22:24 Q. And in this third reassembly step we have highlighted
01:22:32 on the screen, that SIP data moves over from the packet,
01:22:39 the first received packet on the bottom, over here to the

01:22:43 far right, doesn't it?
01:22:46 A. No. We don't move it. We make a copy of that SIP
01:22:49 data.
01:22:49 Q. Okay. Well, let me say it that way then.
01:22:52 The SIP data in this third reassembly step is
01:22:59 copied from the first received packet over here to the top
01:23:03 right in your demonstrative, true?
01:23:06 A. That is true.
01:23:06 Q. And it's copied over without the TCP header, right?
01:23:12 A. That's true.
01:23:13 Q. And it's copied over without the IP header?
01:23:16 A. That's correct.
01:23:17 Q. And it's copied over without the ethernet header?
01:23:20 A. That's correct.
01:23:21 Q. And so at that point in time, as it moves over, it's
01:23:24 just the SIP data itself?
01:23:28 A. We don't move the data.
01:23:29 Q. I'm sorry.
01:23:30 A. Yeah.
01:23:30 Q. At the time it's -- I didn't mean to cut you off.
01:23:33 A. We -- we are copying the SIP data from that -- what's
01:23:37 marked as the first received packet to our new synthetic
01:23:41 ethernet frame.
01:23:42 Q. So at the time it's copied over, it's just that SIP
01:23:44 data?

01:23:45 A. We copy only that SIP data.
01:23:49 Q. And before the reassembly happens, you testified on
01:24:01 direct about something called a TCP state machine. Do you
01:24:04 remember that?
01:24:05 A. Yes.
01:24:05 Q. And the TCP state machine is what's in the GeoProbe
01:24:12 products that processes TCP, fair?
01:24:13 A. That's not entirely accurate. Would you like me to
01:24:18 give a better description?
01:24:19 Q. Well, does it execute a TCP state machine?
01:24:22 A. It executes our implementation of a TCP state machine.
01:24:26 Q. Okay. And there was some source code you were shown --
01:24:33 MR. HURT: May I have the ELMO again,
01:24:35 Ms. Lockhart, please?
01:24:37 Q. (By Mr. Hurt) There was some source code you were
01:24:58 shown, Mr. Curtin, on direct, about TcpSm_processPacket
01:25:19 function. Do you see this on the screen?
01:25:20 A. Yes, I do.
01:25:21 Q. That's the function of the GeoProbe TCP state machine
01:25:24 that processes packets, right?
01:25:26 A. This function processes ethernet packets that have a
01:25:31 TCP header.
01:25:34 MR. HURT: Your Honor, objection. Non-responsive.
01:25:36 THE COURT: Restate your question, counsel.
01:25:47 Q. (By Mr. Hurt) This routine, Mr. Curtin, TcpSm, it

01:25:55 processes packets, right?
01:25:57 A. It does process packets.
01:25:59 Q. And as you testified on direct, one of the things that
01:26:05 happens in this routine is it creates a pointer to the TCP
01:26:10 header, right?
01:26:10 A. That's correct.
01:26:11 Q. And that's at Line 1450 where it says *pTcpHdr?
01:26:18 A. Yes.
01:26:19 Q. And TcpHdr means TCP header, true?
01:26:24 A. Yes, it does.
01:26:25 Q. And the first byte that that pointer points to is the
01:26:31 TCP header; isn't that right?
01:26:34 A. Yes.
01:26:35 Q. NetScout has data sheets that it provides to its
01:27:13 service provider customers; is that right?
01:27:17 A. Yes.
01:27:17 Q. And they explain how -- the parts of the GeoProbe
01:27:23 product, is that -- is that right?
01:27:25 A. Yes.
01:27:29 Q. And those data sheets describe the GeoBlade as a
01:28:35 powerful platform that maximizes capacity and flexibility,
01:28:41 don't they?
01:28:41 A. Yes.
01:28:42 Q. Okay. And capacity means speed, I assume, right?
01:28:50 A. Capacity means packets per second, not exactly speed,

01:28:55 but similar.

01:28:55 THE COURT: You need to speak up, Mr. Hurt.

01:29:00 THE WITNESS: Yes, sir.

01:29:01 THE COURT: Not you, Mr. Curtin, but it won't hurt
01:29:03 if you speak up.

01:29:04 THE WITNESS: Okay. I'll do that. Thank you.

01:29:10 Q. (By Mr. Hurt) And the GeoProbe data sheets also list
01:29:13 as a platform highlight the ability to do real-time packet
01:29:19 processing and dynamic classification, don't they?

01:29:22 A. I don't recall seeing that statement. Could you show
01:29:26 it to me?

01:29:27 Q. Yes. It's on PTX-69 in your witness binder. The third
01:29:34 page down on the right-hand side, there should be something
01:29:38 that says Platform Highlights. Do you see this?

01:29:59 A. Yes, I do see that. Thank you.

01:30:00 Q. And that fourth bullet down on Platform Highlights
01:30:05 says: Real-time packet processing and dynamic
01:30:08 classification.

01:30:10 Is that true?

01:30:16 A. Yes.

01:30:16 Q. Has NetScout -- has there been any effort by NetScout
01:30:20 to change the design of any of the GeoProbe products in
01:30:24 response to this case, Mr. Curtin?

01:30:25 A. No.

01:30:25 Q. And are there any plans to change the design of the

01:30:28 GeoProbe products in response to this case?
01:30:31 A. No.
01:30:34 MR. HURT: Thank you, Your Honor. No further
01:30:35 questions. Pass the witness.
01:30:36 THE COURT: All right. Redirect, Mr. Buresh?
01:30:48 MR. BURESH: Your Honor, we have nothing further.
01:30:50 THE COURT: Then you may step down, Mr. Curtin.
01:30:59 MR. BURESH: Your Honor --
01:31:00 THE COURT: Yes.
01:31:01 MR. BURESH: -- may Mr. Curtin be excused as a
01:31:03 witness?
01:31:03 THE COURT: Any objection from Plaintiff?
01:31:04 MR. HURT: No, Your Honor.
01:31:06 THE COURT: Mr. Curtin, you are excused. You're
01:31:08 free to stay with us. You're also free to leave, sir.
01:31:12 THE WITNESS: Thank you, sir.
01:31:13 THE COURT: Defendant, call your next witness.
01:31:15 MR. BURESH: Your Honor, we call Dr. Scott Dawson.
01:31:22 THE COURT: All right. If you'll come forward,
01:31:26 sir, the courtroom deputy will administer the oath to you.
01:31:49 (Witness sworn.)
01:31:50 THE COURT: Please have a seat at the witness
01:31:55 stand.
01:31:55 SCOTT DAWSON, PH.D., DEFENDANT'S WITNESS, SWORN
01:31:55 DIRECT EXAMINATION

01:31:55 BY MR. BURESH:
01:32:04 Q. Good afternoon, Dr. Dawson.
01:32:05 A. Good afternoon.
01:32:05 Q. Could you state your full name for the record, please?
01:32:09 A. Yes, Scott David Dawson.
01:32:10 Q. And please introduce yourself to the jury and give us a
01:32:13 little background information on yourself.
01:32:15 A. Sure. So my name is Scott Dawson. I grew up in
01:32:18 Dearborn, Michigan, home of Ford -- Ford Motor Company
01:32:24 World Headquarters.
01:32:24 My dad was an automotive engineer, and so he got
01:32:27 me interested in engineering and technology early on.
01:32:30 My mom was an elementary school teacher and later
01:32:33 reading specialist, helping kids that were struggling with
01:32:37 reading.
01:32:37 I now live in Ann Arbor, Michigan, which was about
01:32:40 45 minutes from where I grew up, with my wife of almost 25
01:32:44 years and my daughter who's 13.
01:32:45 Q. Okay. And I'm going to ask you to slow down just a
01:32:48 little bit, please.
01:32:49 A. All right.
01:32:49 Q. And can you describe your educational background
01:32:51 leading up to your Ph.D.?
01:32:52 A. Sure. So I went to the University of Michigan from
01:32:58 1988 to 1998. While there, I received a Bachelor's in

01:33:04 engineering for computer science and computer engineering,
01:33:07 also a Master's in computer engineering, and ultimately a
01:33:10 Ph.D. in computer science.

01:33:11 Q. And what was your area of focus in your doctoral
01:33:16 program?

01:33:16 A. Right. So my dissertation dealt with testing
01:33:21 distributed systems. These are computer systems that
01:33:24 operate on networks and that exchange packets as part of
01:33:30 their operation.

01:33:30 What we were doing was manipulating the packets as
01:33:34 they were exchanged, in order to test whether the protocol
01:33:37 or the system was working. In some cases, we could find
01:33:41 out how it worked or didn't.

01:33:43 Q. And when did you get your Ph.D.?

01:33:48 A. That was in 1998.

01:33:48 Q. And what is your title at NetScout?

01:33:49 A. I am the vice president of engineering.

01:33:51 Q. Can you describe for us what that role entails?

01:33:54 A. Right. So I'm responsible for the Arbor products at
01:33:58 NetScout, and that entails -- I have a team of about 140
01:34:04 people. And we do everything from design, architect, write
01:34:11 the software, check it for quality, and distribute it to
01:34:15 the marketplace.

01:34:15 Q. Are you familiar with the Arbor products, then, that
01:34:18 are at issue in this case?

01:34:18 A. I am.
01:34:19 Q. Could you tell the jury a little bit about what the
01:34:25 Arbor products do at a high level, please?
01:34:27 A. Sure. So what the Arbor products are doing is they
01:34:31 protect our customers and customers of them from something
01:34:36 called distributed denial of service attacks.
01:34:39 Q. What is a distributed denial of service attack?
01:34:43 A. So in one of these attacks, what the attacker is
01:34:46 attempting to do is to take whatever it is they're
01:34:49 attacking, offline.
01:34:51 So whether that be a bank website or a company or
01:34:55 another government or something like that, what they're
01:34:57 trying to do is take that site down so that nobody can
01:35:02 access it.
01:35:02 THE COURT: Dr. Dawson, pull the microphone a
01:35:04 little closer to you, please.
01:35:06 THE WITNESS: Sure.
01:35:06 THE COURT: And as Mr. Buresh asked, please try to
01:35:09 slow down.
01:35:10 THE WITNESS: All right.
01:35:10 THE COURT: All right. Let's continue.
01:35:11 Q. (By Mr. Buresh) How would one of these attackers shut
01:35:18 down a server at like a bank?
01:35:19 A. Sure. So servers have a finite set of resources. So
01:35:26 they might have a finite pool of connections that could be

01:35:29 made to them. And what the attacker does is they try to
01:35:32 occupy all of those connections so that legitimate users
01:35:36 cannot get in and make a connection.
01:35:40 Q. If you were to be at a party with some of your friends
01:35:47 and they ask what do your products do, could you think of
01:35:50 an example of how to describe that to them?
01:35:52 A. Yeah. So to take it out into the real world, one
01:35:56 example I like to use is that of a phone bank. So maybe a
01:36:01 college is trying to get some donations and they open a
01:36:06 phone bank and they've got a whole bunch of people manning
01:36:10 the lines and people are calling in and making donations.
01:36:13 Let's say that's my rival college, and I don't
01:36:16 want them to get donations. So what I do instead is I -- I
01:36:19 get a whole bunch of my friends, and we just continuously
01:36:24 call the phone line and tie up the line so that nobody can
01:36:28 get through. Everybody that's trying to make donations
01:36:31 just gets a busy signal. That's essentially what a denial
01:36:37 of service attack is.
01:36:38 Q. What type of people do these attacks on the -- on the
01:36:40 Internet?
01:36:41 A. So when we started, it was relatively unsophisticated,
01:36:46 might be a kid in their basement or, you know, maybe just
01:36:49 some bad guy trying to play games.
01:36:51 Over the -- over the years since 2001, it's really
01:36:55 evolved. Right now it's anything from governments, nation

01:37:04 states, organized crime, could be corporate competition
01:37:07 trying to take one of their competitors offline.
01:37:10 Q. What do you mean by nation states?
01:37:12 A. So we know, for example, that countries like Iran,
01:37:17 North Korea, China, and Russia all have pretty
01:37:23 sophisticated cyber -- cyber operations, and we know that
01:37:26 there are cases where they'll be doing an invasion.
01:37:31 For example, in 2008, Russia invaded the country
01:37:35 of Georgia, which was next door to them. While they were
01:37:39 doing this physical invasion, they were also doing a denial
01:37:43 of service attack to bring down all of Georgia's
01:37:45 governmental computers.
01:37:45 Q. Can the Arbor products protect against attacks like
01:37:49 that?
01:37:49 A. That's exactly what we do.
01:37:51 Q. Do the Arbor products protect any other sorts of
01:37:54 critical infrastructure?
01:37:55 A. The Arbor products are primarily sold to service
01:38:04 providers, so think like an AT&T or British Telecom. And
01:38:08 those providers use -- use the products to keep their
01:38:11 networks up so that you'd be able to use them. And also
01:38:15 they use them to provide manned services to some of their
01:38:20 customers like banks and others to keep them up.
01:38:23 MR. BURESH: If we could go to Slide No. 1.
01:38:26 Q. (By Mr. Buresh) Dr. Dawson, did you have an

01:38:28 opportunity to assist me in preparing some demonstratives
01:38:31 for your testimony?
01:38:31 A. I did.
01:38:32 Q. And is this the first slide of your demonstratives?
01:38:36 A. It is.
01:38:38 Q. Can you please describe to the jury at a high level how
01:38:47 the Arbor products prevent a denial of service attack?
01:38:50 A. Right. So you can see in this example we've got a bank
01:38:53 who we're protecting using one of the Arbor products, Arbor
01:38:58 APS or Arbor AED.
01:39:03 Out on the other side of our device is basically
01:39:03 the Internet, and there's some green computers there.
01:39:06 Those are regular users just trying to get to the bank
01:39:09 website, maybe pay their bills. And then there's a couple
01:39:12 of red computers with skulls on them. Those are the bad
01:39:15 guys. And they're trying to take the bank offline.
01:39:19 What our device does is it sits inline for all of
01:39:24 that traffic and it inspects that traffic and it determines
01:39:26 on each and every connection that it sees, does this look
01:39:30 good or does it look bad. If it's bad, that traffic would
01:39:33 be dropped. And if it's good, it would be passed through
01:39:37 to the bank.
01:39:38 By doing this inspection and doing this dropping
01:39:40 of bad traffic, we allow the legitimate users access to the
01:39:44 bank while the attackers basically are ineffective at

01:39:48 attacking the bank.
01:39:49 Q. And Dr. Dawson, the -- the green traffic that's going
01:39:52 through to the bank; do you see that?
01:39:53 A. Yes.
01:39:55 Q. What would happen to that green good traffic if the
01:40:00 Arbor products changed the headers of the packets or
01:40:05 converted the packets?
01:40:06 A. Right. So the header of the packet is basically like
01:40:09 the address on an envelope. If we were to change the
01:40:12 packets as they're flowing through our device, they would
01:40:15 never make it to the bank.
01:40:19 Q. I understand there are three products that fall under
01:40:23 the Arbor umbrella; is that correct?
01:40:25 A. That's correct.
01:40:26 Q. What are those products?
01:40:27 A. They are Arbor APS, Arbor AED, and Arbor TMS.
01:40:34 Q. Did you work on any products in the denial of service
01:40:38 space before those products?
01:40:39 A. I did. Around 2001 I worked on a product at Arbor
01:40:44 called Peakflow DoS, which was our initial denial of
01:40:47 service detection product.
01:40:47 Q. What was the business environment like in 2001?
01:40:50 A. Well, we had started in late 2000, and so we were in
01:40:55 the process of building the technology, building a product,
01:40:59 proving it would work, trying to acquire customers, and

01:41:02 trying to make sure that the things that we were building
01:41:05 would work for them. So it was very much a scramble.
01:41:09 Q. So how did the dot-com bust impact that scramble, if
01:41:13 you will?
01:41:14 A. I would say it didn't. Because we were providing
01:41:18 infrastructure, availability, and our target customers were
01:41:22 things like big Internet providers, there was a big need
01:41:26 for DDoS detection at that time, so I think the dot-com
01:41:31 situation didn't really affect us.
01:41:32 Q. What about the 9/11 terrorist attacks and the recession
01:41:38 that followed thereafter, what impact did that have on your
01:41:41 business?
01:41:41 A. Right. So I would say again little to no impact. I
01:41:47 think, if anything, the 9/11 attacks kind of highlighted
01:41:50 for everybody the experience of infrastructure,
01:41:53 availability, security, and things like that. So I think,
01:41:59 you know, it was neutral to positive for us.
01:42:02 Q. When did you start working on developing the Arbor
01:42:06 products that are at issue here in this case?
01:42:09 A. Right. So we began TMS development around 2005.
01:42:15 Q. Is that development ongoing today?
01:42:20 A. It is.
01:42:21 Q. Focussing in on that original 2005 time frame, what was
01:42:29 your role in the project?
01:42:34 A. So shortly after the project started, I became the

01:42:34 manager for the team that was developing the TMS product.
01:42:38 Q. And did your team write software for the Arbor
01:42:42 products?
01:42:42 A. We did. We used some software that we already had on
01:42:47 other products, in terms of operating systems and
01:42:50 base-level functionality.
01:42:51 But in terms of the packet processing, that
01:42:54 determines right from wrong, sort of in this case, on the
01:42:58 screen, that was all developed in-house.
01:43:01 Q. Was that from scratch?
01:43:02 A. Yes.
01:43:03 Q. How long did that take?
01:43:04 A. About a year and a half before we got the first version
01:43:07 to market.
01:43:10 Q. How many people were involved?
01:43:13 A. 10.
01:43:20 Q. Prior to this litigation, had you ever heard of
01:43:24 Mr. Edward Balassanian?
01:43:25 A. I had not.
01:43:27 Q. What about a company named Implicit?
01:43:30 A. No.
01:43:30 Q. A company named BeComm?
01:43:33 A. No.
01:43:33 Q. Prior to this litigation, did you have any awareness of
01:43:39 the patents at issue in this case?

01:43:40 A. I did not.

01:43:44 Q. Besides blocking the malicious or bad traffic like
01:43:49 we're seeing on the screen here, what were some of the
01:43:51 other goals of the Arbor products as you were developing
01:43:55 them?

01:43:55 A. So the primary goal was performance. We needed to be
01:43:58 able to keep up with the networks that we were connected
01:44:01 to.

01:44:03 Q. Did you have a particular performance goal for the
01:44:06 Arbor products?

01:44:07 A. So at that time, common network link was about one
01:44:12 gigabit per second, and now those network links are up to
01:44:17 100 gigabits per second. Our fastest product today handles
01:44:22 400 -- or 400 gigabits per second of traffic.

01:44:26 Q. Can you give us a sense of how fast 400 gigabits of
01:44:32 traffic is?

01:44:32 A. Yes. So a typical home computer connection or cable or
01:44:39 something like that is probably in the range of 40 megabits
01:44:43 per second. 400 gigabits per second is 10,000 times faster
01:44:48 than that.

01:44:54 Q. What about in terms of packets per second, we've been
01:44:56 hearing that. Can you tell us how many packets per second
01:44:58 400 gigabits per second is?

01:45:00 A. Sure. So it varies based on the packet sizes, but if
01:45:05 you use sort of an average packet size, that'd be about 80

01:45:10 million packets per second.

01:45:14 Q. Is speed important to the Arbor products?

01:45:16 A. It is.

01:45:17 Q. Why is that?

01:45:18 A. So the main thing is that any traffic that enters our 01:45:22 device, we have to inspect. And what that means is that if 01:45:27 we can't keep up, if we're not fast enough to keep up, then 01:45:32 we will not be able to inspect every packet that we see or 01:45:36 every connection that we're watching, and we won't be able 01:45:39 to determine good from bad.

01:45:42 Q. What does it mean to drop traffic?

01:45:45 A. It simply means that you don't send that traffic on to 01:45:48 the intended destination.

01:45:50 Q. What impact does that have on the bank?

01:45:53 A. So the impact on the bank of dropping bad traffic is 01:45:57 that the resources of, say, the banking web server, those 01:46:02 open connections that you're able to maintain, those will 01:46:05 not be occupied by bad traffic. And what that allows is 01:46:09 that the good traffic would get through and be able to do 01:46:12 its business.

01:46:14 Q. If you can't keep up with processing, do you 01:46:17 inadvertently drop good traffic?

01:46:20 A. That's what would happen, yes.

01:46:21 Q. And what would be the impact on the bank of dropping 01:46:24 good traffic?

01:46:25 A. Well, then their legitimate customers would not be able
01:46:33 to access the website. That would make our product less
01:46:36 useful to them.
01:46:37 Q. How did you design the Arbor products to accomplish
01:46:41 your speed goals and reach 80 million packets per second?
01:46:44 A. Yeah, so the main thing is being very careful about how
01:46:49 we process the traffic. What we have in terms of
01:46:54 mitigating bad traffic or dropping bad traffic is we build
01:46:59 something called a countermeasure, and we have a series of
01:47:01 these countermeasures in the product that are applied to
01:47:04 all the traffic.
01:47:04 Some of them are easy and some of them are hard,
01:47:10 and so what we do is we order those countermeasures, order
01:47:15 of operation, from easy to hard. And the idea is that, if
01:47:18 I can make a determination with some easy code, then I can
01:47:25 move on and use that processing to do something else, and
01:47:28 I -- I save myself the processing of the harder operation.
01:47:32 So the easy stuff happens upfront. The hard stuff
01:47:36 happens at the back, if at all.
01:47:38 Q. When is that order of processing determined?
01:47:44 A. That's done when we code the product before we ship it
01:47:48 to customers.
01:47:50 Q. Are there processing paths in the Arbor product?
01:47:56 A. Yes.
01:47:57 MR. DAVIS: Objection, Your Honor. May we

01:47:59 approach?
01:47:59 THE COURT: Approach the bench.
01:48:07 (Bench conference.)
01:48:07 THE COURT: All right. Mr. Davis, what is it?
01:48:09 MR. DAVIS: He's using claim language now of
01:48:11 processing paths. And I know maybe there were no
01:48:15 objections to the other witnesses, but I'm going to object
01:48:17 to this witness testifying as an expert, testifying as to
01:48:20 whether or not there are processing paths in the accused
01:48:23 products.
01:48:23 That's a claim term. If he can show that that
01:48:27 word is used in the literature somewhere, that's fine, but
01:48:30 it's not. He's just substituting the claim language.
01:48:35 THE COURT: What's the Defendant's response?
01:48:37 MR. BURESH: The words processing paths do not
01:48:41 appear in the claims, and it is, in fact, the only way to
01:48:44 describe how you process packets along a software path.
01:48:48 I'm not sure what you want me to do with it than use plain
01:48:52 and ordinary meaning since that term wasn't construed
01:48:53 whatsoever. We understand it would have a plain and
01:48:56 ordinary meaning.
01:48:56 MR. DAVIS: I'd be happy to get the claim language
01:49:00 out --
01:49:00 THE COURT: You gentlemen are talking to me,
01:49:02 you're not talking to each other.

01:49:03 MR. DAVIS: The claim uses the word path,
01:49:06 Your Honor. It is one of the two central questions in this
01:49:08 case. It says the -- the element is create a path.
01:49:15 MR. BURESH: And that claim has not been
01:49:18 construed, which means it has a plain meaning.
01:49:20 THE COURT: I'm not going to sustain the
01:49:22 objection. He's entitled to testify from his own personal
01:49:25 knowledge what the products he's familiar with and that his
01:49:27 company sells contain.
01:49:33 Now, if it goes beyond that where he starts to
01:49:37 explain what language like paths means, that's a different
01:49:40 matter, but if he identifies the product that he has
01:49:45 personal knowledge of as either having this attribute or
01:49:49 that attribute, that falls squarely within his personal
01:49:53 knowledge and does not call for an expert opinion.
01:49:55 At this point, I'm going to overrule the
01:49:58 Plaintiff's objection, but I want to put the Defendant on
01:50:02 notice, I do not expect this witness to start giving
01:50:05 opinions about the meaning of language that happens to be
01:50:08 in the claims. That, I think, would be improper, and I'm
01:50:11 not going to permit that. All right?
01:50:13 MR. BURESH: Yes, Your Honor.
01:50:14 THE COURT: Let's proceed.
01:50:16 (Bench conference concluded.)
01:50:29 THE COURT: All right. Let's proceed.

01:50:30 Q. (By Mr. Buresh) Does the Arbor product include
01:50:32 processing paths?
01:50:33 A. It does, yes.
01:50:33 Q. And when are those paths created in the Arbor products?
01:50:36 A. Those are created when we write the software before we
01:50:40 ship it to customers.
01:50:41 Q. When is the arrangement of the processing paths set in
01:50:48 the Arbor products?
01:50:49 A. It's set when we code the product before we ship it to
01:50:52 customers.
01:50:53 Q. How does NetScout define the order of the processing
01:50:57 path in the Arbor products?
01:50:59 A. So, again, we're -- we're organizing those processing
01:51:05 paths so that we're doing the easy things upfront and the
01:51:10 hard things later.
01:51:11 MR. BURESH: If we could go to the next slide,
01:51:16 please.
01:51:16 Q. (By Mr. Buresh) What is this depicting, Dr. Dawson?
01:51:19 A. This shows some processing paths that could be taken
01:51:23 through the product.
01:51:24 Q. Can you describe an example?
01:51:28 A. Yeah, an example would be -- I could get an ethernet
01:51:33 frame into the product. I could determine that that was an
01:51:40 IP -- contained IP data. I might further then look into
01:51:44 the IP header and determine that that contains TCP data, in

01:51:50 which case it would follow the TCP module. Then it would
01:51:54 drop through a series of countermeasures here that might
01:51:58 determine that the traffic is good or bad.
01:52:01 If it's bad, it would go in the trash. We would
01:52:04 simply not send it on.
01:52:06 If it was good, it could continue through.
01:52:10 Depending on the application of -- of this -- that's
01:52:13 contained in this packet, we might apply
01:52:16 application-specific countermeasures to it, again, which
01:52:20 could cause it to be dropped. Or if it wasn't, it would be
01:52:24 transmitted through to the destination.
01:52:26 Q. And what are these countermeasures doing?
01:52:29 A. So that's code that's looking for evidence of good or
01:52:34 bad. In some cases, it's as simple as we know that the
01:52:39 source addresses on the network are bad guys, and we can
01:52:42 simply drop them.
01:52:44 In some cases, it might be rate-based. If
01:52:48 somebody fetches a website 500 times a second, that's not
01:52:52 normal. You know, a normal person would sit in front of a
01:52:56 computer and read the website before clicking again.
01:53:00 In some cases, it might be more complex, like
01:53:03 there's a certain string somewhere in the application that
01:53:08 we know means it's bad traffic.
01:53:10 Q. And what is the order of the countermeasures applied in
01:53:14 the Arbor products?

01:53:14 A. Right. So, again, we're going from things that are
01:53:18 easy to test, to things that are harder to test. So the --
01:53:23 the example of the source we just know is a bad source,
01:53:27 that's something that's relatively easy. That would happen
01:53:30 very early on up here. Something like checking specific
01:53:34 contents of, say, a web packet might be more hard. That
01:53:40 might happen down here.
01:53:41 Q. What about the second set of countermeasures, are those
01:53:45 in an order, as well?
01:53:46 A. They are.
01:53:47 Q. And what is the order there?
01:53:48 A. The order is still from easier things to do, to harder
01:53:53 things to do, although these things in general are harder
01:53:55 than some of the things further up the chain.
01:53:58 Q. Now, is this depiction simplified?
01:54:00 A. It's simplified, yes, but it's -- it's a pretty
01:54:03 accurate depiction of what actually happens in the code.
01:54:05 Q. In the processing paths that are depicted on Slide 3,
01:54:09 when are those created?
01:54:11 A. Those are created when we -- when we compile that
01:54:16 product together and before we ship it to customers.
01:54:20 Q. When in time would that relate to, in terms of actual
01:54:24 packet processing? Is it before?
01:54:25 A. It would be long before.
01:54:28 Q. When you go from -- as we see this IP module, you see

01:54:44 that?
01:54:45 A. Yes.
01:54:45 Q. Into the UDP and the TCP modules, do you see those?
01:54:50 A. Yes.
01:54:50 Q. Does the software make a choice about which path to
01:54:53 follow there?
01:54:54 A. Software doesn't make a choice. The contents of the
01:54:56 packet will walk through a series of if/then/else
01:55:02 statements in the code, and it would say something like, if
01:55:04 this is UDP, execute the UDP processing code. If this is
01:55:11 TCP, execute the TCP processing code.
01:55:14 Q. And, again, make sure you're speaking up.
01:55:17 A. Yeah, sorry.
01:55:18 Q. It's okay.
01:55:19 Are the paths depicted here hard-coded in the
01:55:22 Arbor software?
01:55:23 A. They are, yes.
01:55:24 Q. What is the degree of changeability or modifiability of
01:55:29 these paths in the Arbor products?
01:55:31 A. Only back at the factory, so to speak. Users are not
01:55:36 able to make modifications.
01:55:38 Q. Is there any run time modifications?
01:55:40 A. There is not.
01:55:41 Q. Now, the connections between these modules, like the
01:55:56 ethernet to IP module --

01:55:57 A. Uh-huh.
01:55:57 Q. -- how are those connected together?
01:56:00 A. So those are connected in code that looks at the
01:56:03 contents of the ethernet -- ethernet header on the ethernet
01:56:08 packet.
01:56:08 And depending on what is contained there, if it
01:56:13 was, for example, an IP -- if it contained IP data, then it
01:56:18 would execute the code that would process the IP data.
01:56:22 Q. And I believe in a notebook you have source code
01:56:25 available to you if you need it.
01:56:26 A. Okay.
01:56:27 Q. But if I -- if I ask you to show me the code that
01:56:30 connects the ethernet to IP module, would you be able to do
01:56:34 that?
01:56:34 A. I would, yes.
01:56:36 MR. BURESH: Mr. Palisoul, can you please pull up
01:56:39 in that split screen format DX-453 at 108 to 109? Thank
01:56:47 you.
01:56:47 Q. (By Mr. Buresh) Dr. Dawson, can you describe for the
01:57:01 jury what's happening in this source code?
01:57:04 A. Right. So what's happening here is we're looking at
01:57:07 the ethernet header, and you can see we're switching on the
01:57:13 protocol up near the top here at 316 -- Line 316,
01:57:18 highlighted in yellow. If that's of type IP, then you can
01:57:24 see -- drop down a few lines, we're going to call the IP

01:57:27 processing routine with that data.

01:57:32 Q. Are these what's called instructions in the source
01:57:34 code?

01:57:34 A. That's right. This is source code. These are the
01:57:36 instructions. Case statement is essentially like what I
01:57:40 said before, if/then/else, so if either type IP call PCIP
01:57:49 process.

01:57:49 Q. What happens in a computer when instructions like this
01:57:53 are executed?

01:57:53 A. It's simply calling the next set of code, and it's
01:58:00 running the next set of code.

01:58:00 Q. Are instructions flexible?

01:58:01 A. They are not.

01:58:08 Q. In the Arbor products, what degree of flexibility is
01:58:13 there in reordering the paths that we're looking at on the
01:58:16 left-hand side of the screen?

01:58:17 A. There's no flexibility in reordering those. We would
01:58:22 have to change the source code back at the factory, so to
01:58:25 speak, for that to happen.

01:58:27 Q. Between the other modules, are there similar
01:58:34 instructions that connect those modules?

01:58:37 A. That's right.

01:58:38 MR. BURESH: If we could advance the slide,
01:58:39 please.

01:58:39 Q. (By Mr. Buresh) I want to do another example between

01:58:42 IP module and TCP module. Do you see that?
01:58:44 A. I do.
01:58:45 Q. And can you show me the -- similarly show me the source
01:58:49 code that we connect those two together?
01:58:50 A. I can.
01:59:03 MR. BURESH: Mr. Palisoul, if you could pull it up
01:59:05 in split screen mode to DTX-454 at 730.
01:59:12 A. Right. So you can see here we're looking at the IP
01:59:16 header. And we're switching on the protocol. And
01:59:21 highlighted on Line 141, if that's a protocol TCP, then
01:59:28 we're going to call a function called new_tcp_segment with
01:59:35 the packet.
01:59:36 Q. And is that a fixed instruction?
01:59:37 A. It is, yes.
01:59:39 Q. If we walk through each of the various connections
01:59:42 depicted on these processing paths, are they all connected
01:59:48 by similar instructions?
01:59:49 A. They are, yes.
01:59:49 Q. When are the processing paths in the Arbor product
01:59:56 created?
01:59:56 A. Those are created when we write the software before we
02:00:00 ever ship it to customers.
02:00:02 Q. How many lines of code does that take?
02:00:04 A. It's a lot. I would say across all Arbor products, we
02:00:08 probably have in the low millions of lines of code.

02:00:12 Q. What do you have to do if you want to add a new pipe or
02:00:16 a new processing routine?
02:00:18 A. So what we would do, typically the -- the new stuff
02:00:22 that we add is new countermeasures for a new attack that
02:00:25 might show up out there.
02:00:27 What we do is we go and we try to understand what
02:00:31 the service is or what the application is that could be
02:00:34 attacked. And then we try to figure out how could we tell
02:00:38 if we're looking at attack traffic or good traffic. And we
02:00:43 write some new code that would try to determine is this
02:00:46 good traffic or bad traffic.
02:00:48 Then we would figure out kind of how expensive
02:00:51 that is to run that code or how hard it is or how easy it
02:00:55 is. And we would place that at the -- at the sort of
02:00:59 appropriate place in our processing path based on that
02:01:03 difficulty level of that code.
02:01:05 And then we would create a new version of the
02:01:09 software, and we would ship it out to customers that have
02:01:11 our devices. They would install it on those devices, and
02:01:14 then they would be able to use that code.
02:01:17 THE COURT: All right. I'm going to interrupt
02:01:18 here.
02:01:19 Ladies and gentlemen, we're going to take a short
02:01:21 recess before we continue.
02:01:22 You can simply leave your notebooks in your

02:01:24 chairs. Be sure to follow all the instructions I've given
02:01:28 you regarding your conduct, including, of course, as you
02:01:30 would expect me to remind you, not to discuss the case with
02:01:33 each other. And we'll be back shortly to continue.

02:01:36 The jury is excused for recess.

02:01:38 COURT SECURITY OFFICER: All rise.

02:01:56 (Jury out.)

02:01:56 THE COURT: The Court stands in recess.

02:01:56 (Recess.)

02:15:54 (Jury out.)

02:15:54 COURT SECURITY OFFICER: All rise.

02:16:02 THE COURT: Be seated, please.

02:16:48 All right. Counsel, are you prepared to continue
02:16:54 with your examination of the witness?

02:16:56 MR. BURESH: I am, Your Honor.

02:16:57 THE COURT: You may return to the podium.

02:16:58 Let's bring in the jury, Mr. Brooks.

02:17:24 COURT SECURITY OFFICER: All rise.

02:17:24 (Jury in.)

02:17:26 THE COURT: Please be seated.

02:17:37 We'll continue with the Defendants' direct
02:17:42 examination of the witness.

02:17:43 You may proceed, counsel.

02:17:44 MR. BURESH: Thank you, Your Honor.

02:17:46 Mr. Palisoul, if we could pull the slides up where

02:17:50 we left off.

02:17:51 Q. (By Mr. Buresh) And, Dr. Dawson, again, looking at the
02:18:07 demonstrative in front of you, what ability does the Arbor
02:18:11 product have to add a new processing path after receiving a
02:18:17 first packet?

02:18:18 A. There is no ability to do -- to do so.

02:18:21 Q. What ability does the Arbor products have to add a new
02:18:29 connection between pipes or processing paths after a packet
02:18:32 has been received?

02:18:32 A. No ability to do so.

02:18:39 Q. What ability do the Arbor products have to reorder the
02:18:42 processing after a packet has been received?

02:18:44 A. That is not possible.

02:18:46 Q. On the demonstrative in front of you now, this is
02:19:00 depicting an Arbor in a network receiving packets. Do you
02:19:05 understand that?

02:19:05 A. I do.

02:19:05 Q. Okay. Looking at the packet on the lower left-hand
02:19:09 portion of the screen, in what order will the processing
02:19:15 paths be called to process this packet?

02:19:18 A. So, first, we will process the ethernet data in the
02:19:23 packet. Then we'll move on and process the IP data in the
02:19:26 packet. Then we would process the TCP data in the packet.
02:19:30 And, finally, moving down, we would process in the
02:19:35 application-specific data in the packet.

02:19:37 Q. What ability do the Arbor products have to change that
02:19:39 order after receiving this packet?
02:19:41 A. No ability to do so.
02:19:44 Q. When you -- assuming a supported protocol, what happens
02:19:53 in the Arbor products in order to execute the processing
02:19:59 path?
02:19:59 A. So in a supportive protocol, we would be executing code
02:20:04 that would be looking at the data in the -- in these -- in
02:20:10 the packet, and it would be calling functions through the
02:20:13 path of code based on what the data is in the packet.
02:20:16 So, for example, the ethernet determines that it's
02:20:20 an IP packet, we process the IP data in the packet, and we
02:20:24 move on and process the TCP data in the packet. And then
02:20:27 we do any application-specific processing.
02:20:32 MR. BURESH: Mr. Palisoul, can you advance the
02:20:34 animation, please?
02:20:38 Q. (By Mr. Buresh) Now, in the lower left-hand portion of
02:20:40 this, you see the packet depicted there?
02:20:42 A. I do.
02:20:42 Q. If this is an unsupported protocol in the Arbor
02:20:46 products, what happens?
02:20:48 A. So for an unsupported protocol, what would happen is we
02:20:53 can't do any application-specific processing because we
02:20:56 don't understand the application.
02:20:58 We will still process the ethernet header, and

02:21:03 then decide to process the IP data in the packet, and
02:21:06 decide to process the TCP data in the packet, and some of
02:21:09 those countermeasures that are sort of up in this area
02:21:12 could still be run on the data in the packet. But we won't
02:21:16 be able to do any application-specific things because -- if
02:21:21 we don't understand the application.
02:21:22 Q. What would you have to do with respect to the Arbor
02:21:27 products to be able to process this type of packet we have
02:21:29 depicted here?
02:21:30 A. So if we -- if we had an application that wasn't
02:21:35 understood by the product, what we would have to do is go
02:21:39 back to the factory and basically write new code. We'd
02:21:43 have to analyze that protocol or that service. We'd have
02:21:49 to understand and just -- and try to figure out if there
02:21:53 was a way to tell good from bad, how do we find the DDoS
02:21:57 attack within this protocol.
02:21:58 And then what we would do is write some new code.
02:22:02 We would insert that code into the series of
02:22:05 countermeasures in an order from sort of easy to hard. If
02:22:09 it was one of the harder ones, it would end up near the
02:22:12 end. And then we would create a new version of the
02:22:14 software, and we would ship that to customers, and they
02:22:19 would deploy that on their machines.
02:22:22 Q. Now, let's assume this packet in front of us here is
02:22:25 from a bad actor or the nation's state that's doing the

02:22:29 attack.
02:22:29 A. Uh-huh.
02:22:30 Q. What happens in that context?
02:22:34 A. It's from the nation's state, but it's something that
02:22:37 we don't understand.
02:22:38 Q. No, in this context -- let's just say it's this
02:22:41 ethernet packet with HTTP data.
02:22:43 A. Okay. In that case, if this was a bad -- a bad
02:22:49 connection of HTTP, it's coming from a nation's state and
02:22:52 we're able to figure out that this is bad traffic, we would
02:22:56 simply not send that traffic to the destination. So you
02:22:59 could see it ends up over here in the trash. There's no
02:23:05 actual trash can. What happens is we just don't send it
02:23:07 on, and that means it's lost to the network.
02:23:10 Q. When the Arbor products throw bad traffic in the trash,
02:23:18 does that change the processing path in any way?
02:23:20 A. It does not.
02:23:21 Q. How do you know?
02:23:22 A. Because we coded it.
02:23:26 Q. To what extent are the processing paths in the Arbor
02:23:31 products fixed?
02:23:31 A. They're fixed.
02:23:38 MR. BURESH: If you could advance the slides,
02:23:41 please.
02:23:41 Q. (By Mr. Buresh) Here we have an ethernet packet with

02:23:44 XCAP data; do you see that?
02:23:46 A. I do.
02:23:51 MR. BURESH: Your Honor, for the next period of
02:23:53 time, I'm going to need to seal the courtroom.
02:23:57 THE COURT: All right. Based on counsel's
02:23:59 request, I'll order the courtroom sealed to protect
02:24:02 confidential or proprietary information.
02:24:05 Those present not subject to the existing
02:24:07 protective order that's been entered by the Court in this
02:24:09 case should excuse themselves and remain outside until the
02:24:12 courtroom is unsealed.
02:24:12 For the record, the courtroom is sealed.
02:24:12 (Courtroom sealed.)
02:24:12 (This portion of the transcript is sealed and
02:24:12 filed under separate cover as Sealed Portion
02:24:17 No. 6.)
02:24:17 (Courtroom unsealed.)
02:33:54 THE COURT: All right. Let's proceed.
02:33:57 Q. (By Mr. Buresh) What type of packets are received in
02:34:07 the Arbor products?
02:34:07 A. Ethernet packets.
02:34:13 Q. What ability do the Arbor products have to change the
02:34:16 header structure of those packets?
02:34:18 A. They don't change the headers of the packets.
02:34:20 Q. Why not?

02:34:21 A. Because, again, changing the header is like changing
02:34:24 the address on an envelope. If we were to change the
02:34:26 header, then any traffic that we decide to send on to the
02:34:29 destination wouldn't make it to the destination.
02:34:34 MR. BURESH: If we could pull up the next slide,
02:34:36 please.
02:34:36 Q. (By Mr. Buresh) Looking at this next demonstrative,
02:34:51 Dr. Dawson, could you describe for us what happens when a
02:34:55 packet is received in the Arbor products?
02:34:56 A. Right. So the first thing that happens is the packet
02:35:00 is copied off the network and into system memory somewhere.
02:35:04 We then create these bookkeeping objects called an abuf and
02:35:09 a pbuf that help us to keep track of that packet and where
02:35:12 it is in memory.
02:35:14 Q. What part of the packet is placed in memory?
02:35:16 A. The entire ethernet packet is placed in memory.
02:35:20 Q. What happens to the ethernet packet while it's in
02:35:24 memory and being processed?
02:35:26 A. Nothing. It's just simply looked at.
02:35:28 Q. What is the outermost header of the packet depicted on
02:35:35 this slide?
02:35:35 A. That is an ethernet header.
02:35:38 Q. How do you know that?
02:35:39 A. Because we receive the full ethernet frame, including
02:35:44 the header, and we place that in memory.

02:35:47 Q. Now, this structure, abuf and pbuf, does it point to
02:35:53 the entire packet?
02:35:54 A. So the abuf points to the entire packet. You can see
02:35:58 the head -- head pointer here points to the beginning. And
02:36:04 the tail pointer here points to the very end of the data in
02:36:05 the packet.
02:36:06 Q. And what does the pbuf do?
02:36:09 A. The pbuf is a way of quickly indexing into the other
02:36:12 layers of the packets, so we have the Layer 3 header
02:36:18 pointer and the Layer 4 header pointer, and then finally a
02:36:22 pointer to the application payload there.
02:36:22 Q. To what extent do these pointers change the packet?
02:36:26 A. The pointers don't change the packet at all. They're
02:36:28 just a quick way of indexing into the packet, like a
02:36:32 bookmark in a book.
02:36:35 Q. Could you show us these structures, the abuf and the
02:36:39 pbuf, where they're created in the source code at the Arbor
02:36:42 products?
02:36:42 A. I can, yes.
02:36:43 MR. BURESH: Mr. Palisoul, if you could, please,
02:36:47 in a split screen format, pull up DX-454 at 726. Thank
02:36:58 you.
02:36:58 Q. (By Mr. Buresh) And what do we see here, Dr. Dawson?
02:37:00 A. So what we see here is the beginning of the definition
02:37:04 of the abuf structure definition. You can see that there's

02:37:08 an acap reference. That's the reference to the interface
02:37:14 that we receive the packet off of. And then you have a
02:37:14 pointer to the head of the packet, and you have a pointer
02:37:17 to the tail of the packet.

02:37:18 Q. Does the abuf enable access to the entire packet?

02:37:22 A. It does.

02:37:25 MR. BURESH: If we advance the slide, please.

02:37:38 Q. (By Mr. Buresh) What are we seeing here?

02:37:40 A. So here we're seeing the beginning of the pbuf data
02:37:43 structure, and -- and there's a little bit of a gap there,
02:37:45 but you can see there's a pointer to the Layer 3 header
02:37:50 and, again, to the Layer 4 header, again, basically
02:37:54 bookmarks into the packet at those places in the packet.

02:37:59 Q. So how do the processing routines along the processing
02:38:02 path, how do they use these structures?

02:38:04 A. So what the processing routines do is when they want to
02:38:07 look at one portion of the packet or another, they're able
02:38:09 to use these pointers to reference into that portion of the
02:38:12 packet and analyze that part of the data.

02:38:15 So, for example, when I'm processing an IP header,
02:38:18 I use that L3 header pointer to look at the IP data and
02:38:23 make whatever decisions -- you know, basically inspect that
02:38:27 data at that point.

02:38:30 MR. BURESH: And, for the record, this is DX-454
02:38:32 at 733 and 734.

02:38:35 Q. (By Mr. Buresh) If I ask you to show me an example of 02:38:41 a processing -- processing routine being past the abuf or 02:38:48 pbuf, could you do that for me?

02:38:50 A. I could, yes.

02:38:51 MR. BURESH: Mr. Palisoul, could you, again, in 02:38:53 split screen, pull up DX-453 at 191?

02:39:07 Q. (By Mr. Buresh) What are we seeing here, Dr. Dawson?

02:39:09 A. So what we're seeing here is a call to a function 02:39:12 called new_tcp_segment. This is called any time when we 02:39:17 have a packet that contains some TCP data. And we call 02:39:21 that function, and we give it the entire abuf so that it's 02:39:25 able to do whatever processing would be done on that TCP 02:39:29 data.

02:39:30 Q. Is new_tcp_segment a TCP -- part of the TCP processing 02:39:37 routine?

02:39:37 A. It is.

02:39:37 Q. And why is it past the entire -- or, I'm sorry, why is 02:39:42 it past the abuf?

02:39:43 A. So it's past the abuf because we wanted to have access 02:39:47 to all of the -- the entire ethernet packet that we 02:39:49 received off the network.

02:39:51 In fact, in the case of new_tcp_segment, one of 02:39:57 the first things it does is it does an IP check zone on a 02:40:01 packet which actually has to reference back into the IP 02:40:03 header data. And so by having the full abuf, it's able to

02:40:06 do any of that processing that it needs to do.

02:40:09 Q. To what extent do the Arbor products ever remove or
02:40:12 change a header on a packet?

02:40:14 A. They don't ever remove or change headers. We always
02:40:17 leave the entire ethernet frame sitting in memory and --
02:40:21 and only look at different portions of it.

02:40:24 Q. To what extent do the Arbor products convert packets in
02:40:27 any way?

02:40:28 A. They don't convert anything. They simply leave the
02:40:31 packet there in memory and look across that packet and
02:40:37 process what -- what they need to.

02:40:40 Q. Do the Arbor products ever process a packet that has
02:40:43 TCP as the outermost header?

02:40:45 A. They do not.

02:40:59 MR. BURESH: If we could bring up Dr. Almeroth's
02:41:01 Slide 101, please.

02:41:03 Q. (By Mr. Buresh) I know you haven't seen this slide
02:41:15 before, Dr. Dawson, but just focusing on the source code,
02:41:19 do you recognize that?

02:41:20 A. Yes, I do.

02:41:21 Q. What is it?

02:41:22 A. This is the TCP processing, the file that basically
02:41:28 defines all the TCP processing in the product.

02:41:33 Q. When this process is being executed --

02:41:38 A. Uh-huh.

02:41:39 Q. -- does that create a packet that has TCP as an
02:41:44 outermost header?
02:41:44 A. It does not.
02:41:45 Q. Does this processing routine that we're looking at here
02:41:57 ever create a representation of a packet that has TCP as an
02:42:00 outermost header?
02:42:01 A. We don't create any representations of packets. We
02:42:05 always have the full packet, and we're simply indexing into
02:42:09 those packets.
02:42:14 MR. BURESH: If we could pull up Slide 164 from
02:42:17 Dr. Almeroth's testimony.
02:42:20 Q. (By Mr. Buresh) Again, I know you haven't seen this
02:42:22 slide, but focusing in on source code, can you describe
02:42:25 what we're looking at in the code here?
02:42:27 A. Yeah. So this part of code is part of the code that we
02:42:32 use to decode HTTP traffic or web traffic. And one of the
02:42:38 things that we need to do when we're doing that is
02:42:42 sometimes the data in an HTTP stream will cross several
02:42:47 packets, and when that happens, it's very hard to operate
02:42:50 on that fragmented data.
02:42:54 And so what we do is aggregate it up into a single
02:42:58 buffer that we can then more easily operate on. And so
02:43:02 what's circled here on Line 110 is the HTTP_HEADER buffer
02:43:13 that we would be using that area of memory to pull together
02:43:15 potentially data from multiple packets into one area so we

02:43:18 could more easily operate on it.
02:43:23 Q. Does this source code create a packet or a
02:43:26 representation of a packet that has the application layer
02:43:30 as the outermost layer?
02:43:32 A. No. This is just one of several fields in the HTTP
02:43:37 header or in the HTTP stream that we might look at. You
02:43:41 can see there are others like requests or response or
02:43:44 things like that.
02:43:45 Q. Does this processing change the packet in any way?
02:43:49 A. It does not.
02:44:00 MR. BURESH: Your Honor, I pass the witness.
02:44:01 THE COURT: Cross-examination by the Plaintiff?
02:44:03 MR. DAVIS: Yes, Your Honor.
02:44:20 THE COURT: Proceed when you're ready, Mr. Davis.
02:44:22 MR. DAVIS: Thank you, Your Honor.
02:44:22 CROSS-EXAMINATION
02:44:23 BY MR. DAVIS:
02:44:23 Q. Good afternoon, Dr. Dawson.
02:44:24 A. Good afternoon.
02:44:25 Q. How are you?
02:44:25 A. I'm good.
02:44:26 Q. We're never met before, have we?
02:44:28 A. We have not.
02:44:29 Q. Well, it's good to meet you. My name is Bo Davis, and
02:44:31 I represent the Plaintiff in this case, Implicit.

02:44:35 A. Okay.

02:44:43 MR. DAVIS: Now, if I could have DX-464, please,
02:44:45 Mr. Diaz?

02:44:47 Q. (By Mr. Davis) You were asked on direct examination,
02:44:49 Dr. Dawson --

02:44:50 MR. BURESH: Your Honor, may I approach or just
02:44:55 I'm going to request to seal the courtroom. Any time this
02:44:58 document is being used, we need to have it sealed.

02:45:01 MR. DAVIS: Your Honor, may I also request to seal
02:45:03 the courtroom?

02:45:03 THE COURT: Then based on that unanimous request,
02:45:05 I'll order the courtroom sealed.

02:45:07 All present who are not subject to the Court's
02:45:11 protective order in this case should excuse themselves and
02:45:13 remain outside until the courtroom is unsealed.

02:45:17 If you're in the courtroom and you're not subject
02:45:19 to the protective order in this case, you need to excuse
02:45:22 yourselves at this time.

02:45:36 All right. The courtroom is sealed.

02:45:42 (Courtroom sealed.)

02:45:42 (This portion of the transcript is sealed and
02:45:42 filed under separate cover as Sealed Portion
02:45:42 No. 7.)

03:04:26 (Courtroom unsealed.)

03:04:27 THE COURT: And if you'll come forward and be

03:04:31 sworn, Dr. Jeffay.
03:04:51 (Witness sworn.)
03:04:52 THE COURT: Please come around, sir, have a seat
03:04:54 on the witness stand.
03:05:05 All right. Counsel, you may proceed with your
03:05:07 direct examination.
03:05:07 KEVIN JEFFAY, PH.D., DEFENDANT'S WITNESS, SWORN
03:05:07 DIRECT EXAMINATION
03:05:09 BY MR. BURESH:
03:05:09 Q. Good afternoon, Your Honor, Dr. Jeffay.
03:05:12 A. Good afternoon.
03:05:13 Q. Could you please state your full name for the record?
03:05:15 A. My name is just Kevin Jeffay.
03:05:17 Q. And give us a little bit about your background.
03:05:20 A. Sure. I'm a professor of computer science at the
03:05:25 University of North Carolina at Chapel Hill. I've been in
03:05:29 Chapel Hill for now more than half my life, but I'm
03:05:35 originally a mid-westerner from Illinois. I -- as I say,
03:05:39 I've been in Chapel Hill for 30 years, married, have two
03:05:42 kids, both grown and off getting busy with their lives.
03:05:45 Q. Can you describe for us your educational background?
03:05:51 A. Sure. I was raised in sort of a science house, I
03:05:57 guess. My father was a biochemist, and my mom was a -- had
03:06:02 been a scientist. By the time I came around, she was just
03:06:04 a stay-at-home mom.

03:06:05 I had an interest in mathematics, of all things,
03:06:08 as a kid, and I went to school at the University of
03:06:11 Illinois to study mathematics.

03:06:14 While there, that's where I discovered computing.
03:06:18 I worked to go to school, and I worked for the Army Corps
03:06:24 of Engineers, which you may know is the -- it's sort of a
03:06:28 quasi civilian branch of the military that maintains a lot
03:06:31 of the public infrastructure like dams and bridges.

03:06:35 But I worked in a -- what was called the
03:06:38 construction engineering research lab and helped build a
03:06:41 computer system to wash tanks for the Army, and that was a
03:06:44 ton of fun as a -- as a late teenager.

03:06:47 And that got me really interested in computing.
03:06:50 And so when I graduated, I switched to computer science and
03:06:54 went to graduate school to get some credentials in computer
03:06:58 science.

03:06:59 Q. And where did you go to graduate school?

03:07:01 A. I went to graduate school twice. But, ultimately, I
03:07:04 went to the University of Washington, that we heard about
03:07:07 earlier, and received a Ph.D. in computer science.

03:07:09 Q. What did we hear about the University of Washington
03:07:15 earlier?

03:07:15 A. We heard that's where Mr. Balassanian went to school.
03:07:20 And while I was teaching there, I believe we actually
03:07:23 probably overlapped a bit. So I might have actually had

03:07:27 him in class. I don't know.

03:07:29 And we also heard that it was a great -- I think 03:07:32 Dr. -- Mr. Balassanian said it was one of the best computer 03:07:38 science schools in the country, and I certainly would agree 03:07:42 with him.

03:07:42 Q. Did you have any project work or projects that led into 03:07:46 your thesis, or whatever, for your Ph.D.?

03:07:48 A. Yes. I did an operating systems Ph.D. And my project 03:07:54 was joint with a group at Boeing Aerospace. Boeing was 03:07:58 quite big in Seattle then. Boeing had a contract to do 03:08:02 work on what was -- what eventually became the 03:08:07 International Space Station.

03:08:08 And part of my research was to help them write an 03:08:11 operating system for one of the flight control computers 03:08:14 that, as best I know, actually launched. And so it's kind 03:08:17 of neat to think that at least for some amount of time 03:08:20 there was code that was orbiting the earth that I helped 03:08:25 write.

03:08:25 Q. I'm sorry, what year did you get your Ph.D.?

03:08:28 A. That would be 1989.

03:08:31 Q. Okay. What did you do after you obtained your Ph.D.?

03:08:38 A. So I wanted to try my hand at being an academic. I 03:08:42 really liked teaching and I liked research, and so I 03:08:42 accepted a position at the University of North Carolina 03:08:45 and moved all the way from the West Coast to the East

03:08:48 Coast, and I've been in North Carolina ever since -- ever
03:08:53 since 1989.
03:08:53 Q. Do you have a title at the University of North
03:08:57 Carolina?
03:08:57 A. I do. I have a bit of a long title. I am officially
03:09:01 the Gillian Cell Distinguished Professor, Computer Science,
03:09:06 and I have an administrative position, in that I serve as
03:09:08 the chair -- chairman of the Department of Computer
03:09:10 Science.
03:09:10 Q. How long have you been the chairman of the Department
03:09:16 of Computer Science?
03:09:16 A. This is year number 6.
03:09:17 Q. Now, do you actually still teach classes at the
03:09:25 University of North Carolina?
03:09:27 A. I do. I taught an operating systems class this
03:09:32 semester. And while I'm down here, I'm actually grading
03:09:35 final exams.
03:09:36 Q. How do you describe what a packet is to your students?
03:09:41 A. I tell them that a packet is the basic unit of
03:09:46 transmission on the network. And as we've heard here,
03:09:50 several witnesses have talked about it, we use the U.S.
03:09:54 mail system as an example because it's pretty good, that a
03:09:57 packet is like an envelope. It contains a letter. The
03:10:01 letter is the data. And there's some addressing
03:10:03 information on the outside. And that's sort of like a --

03:10:05 like the headers of a packet.
03:10:06 Q. How do the -- how does the addressing information
03:10:13 relate to the headers of a packet?
03:10:16 A. So it's not a perfect analogy, but it's -- it's pretty
03:10:19 good. We -- we tend not to think about this much, but the
03:10:23 lines -- each line on an address on an envelope serves a
03:10:27 separate purpose.
03:10:28 So, for example, when you first mail a letter, all
03:10:30 the post office is looking at is really the zip code
03:10:33 because what they want to know is what's the remote -- the
03:10:36 destination -- postal station that's going to get the
03:10:39 letter. So they look at the zip code.
03:10:42 When it arrives at the destination postal station,
03:10:46 they'll look at -- if there's a four-digit extension on the
03:10:50 zip code, that specifies the -- what's called the carrier
03:10:56 route -- essentially the mail carrier's route. And so that
03:10:57 would sort of be like another level of header, another
03:11:00 level of protocol.
03:11:01 And then once the carrier is on his route, then
03:11:04 they look at the street address. That's what they need to
03:11:06 know to deliver it, so that's sort of like another level of
03:11:10 protocol.
03:11:10 And we really don't think about this, but it -- it
03:11:12 kind of makes sense that if there's a lot of people living
03:11:15 in your home when the mail comes into your home, someone

03:11:18 would look at the top line of the address to see who in the
03:11:21 home to deliver the actual letter to, whether it's me or my
03:11:25 wife or one of the kids.

03:11:26 So each one of those lines you can kind of think
03:11:29 of as a protocol because they serve a separate purpose in
03:11:32 the -- in the overall delivery of the letter.

03:11:34 Q. Do you have any awards or accolades, that sort of
03:11:40 thing, that are in the 1990s, the time frame of -- of when
03:11:44 Mr. Balassanian was doing his work?

03:11:45 A. Yes. In the 1990s, starting like literally around
03:11:54 1990, I got very interested in problems of transmitting
03:11:58 audio and video over the Internet. It -- it seems kind of
03:12:03 quaint today, but -- because it's just so common, but --
03:12:05 but it was a cutting-edge problem in 1990. And I was
03:12:10 building operating systems and designing what are called
03:12:12 transport protocols for delivering audio and video.

03:12:15 And my research group, a bunch of students at the
03:12:20 university and I, we did win a number of awards throughout
03:12:22 the 1990s for work we did in operating systems and network
03:12:28 protocol and development.

03:12:29 Q. Have you won any other awards that are relevant to the
03:12:34 technology that's at issue in this case?

03:12:36 A. Yes. Starting around '97 or so, I got very interested
03:12:41 in issues of trying to understand traffic flowing on
03:12:44 networks because, as we've heard, you know, the Internet

03:12:49 was exploding and being used for all kinds of things. And
03:12:54 we really did not understand the impact that all these
03:12:56 different traffic types had on the Internet.

03:12:58 And so to study Internet traffic, my research
03:13:03 group, we built devices that sit in the middle of the
03:13:07 network and monitor packet traffic. And in our case, we
03:13:10 were monitoring packet traffic on the campus -- the UNC
03:13:15 campus network.

03:13:16 And in 2001, my group won what's arguably the most
03:13:22 prestigious award in computer networking for original
03:13:25 research. It was based on a particular type of monitoring
03:13:27 device that we had built.

03:13:29 Q. Have you been awarded any patents for any of your
03:13:36 research or projects?

03:13:38 A. Well, I'm a named inventor on patents. I -- I don't --
03:13:41 I don't actually get the patents, so they're not awarded to
03:13:44 me. The university takes them. And several of my students
03:13:46 have wanted to apply for patents just so they can be
03:13:50 considered an inventor. And so we've applied for -- four
03:13:56 times, we've applied for patents, and so I'm a named
03:13:58 inventor, I believe, on four -- four patents.

03:13:59 Q. So do you have experience with operating systems that
03:14:07 process network packets?

03:14:08 A. Yes. I've developed those operating systems, and I've
03:14:12 used commercial operating systems for that purpose.

03:14:14 Q. For how long?
03:14:15 A. I guess probably going -- yeah, all the way back to
03:14:23 when I started at -- as a faculty member, ever since I
03:14:27 started UNC, so roughly 30 years.
03:14:30 Q. Are you familiar with the concept of deep packet
03:14:33 inspection?
03:14:34 A. I am.
03:14:35 Q. What about application level inspection?
03:14:39 A. I'm familiar with that.
03:14:40 Q. Are those similar concepts?
03:14:42 A. Yes, they're -- they're similar.
03:14:44 Q. And what do they mean?
03:14:45 A. I guess the simplest way to think about it is, if we go
03:14:49 back to the letter analogy, you get your letter, and what
03:14:51 you want to do is try and understand something about the
03:14:53 data in the letter without necessarily having to open up
03:14:57 the letter. So you're sort of like inspecting into -- into
03:15:00 the letter -- into the envelope to look at whatever is
03:15:04 contained inside that envelope.
03:15:09 Q. When was deep packet inspection or application level
03:15:16 inspection first available?
03:15:17 A. I -- I don't know that I can put a precise date on it,
03:15:21 but easily 30 -- 30-plus years ago.
03:15:24 Q. What about flow-based processing, what is that?
03:15:30 A. Flow-based processing is -- is related to application

03:15:37 inspection. The idea is applications can transmit pieces
03:15:40 of data that are too big to fit in one envelope, so you
03:15:43 have to chop up the data into multiple envelopes, and the
03:15:48 transmission of those multiple envelopes, those multiple
03:15:51 packets, constitutes what we call flow. And so flow-based
03:15:55 processing essentially is looking at groups of packets that
03:15:58 are related to one another, and that's also a fairly old
03:16:02 concept.

03:16:02 Q. When did flow-based processing come about?

03:16:05 A. Again, I don't know that I can put an exact date on it,
03:16:09 but we've heard about the RMON standard, which I certainly
03:16:14 knew of. It's a flow-based processing standard called
03:16:16 remote monitoring. And that dates from the early '90s,
03:16:21 around '91.

03:16:22 Q. Are any of your patents related to flow-based
03:16:28 processing?

03:16:28 A. Yes, yes. I think at least one -- one of them is. One
03:16:33 of them, we have a patent about -- it's not really related
03:16:38 to the issues here other than from a mechanism standpoint.
03:16:41 It is about processing packet headers to understand
03:16:45 properties of traffic.

03:16:46 Q. Is that concept of processing packet headers to
03:16:50 understand what's in the traffic, has that been around for
03:16:52 a while, as well?

03:16:53 A. Yes, that's the way we always understood -- that's the

03:16:57 means by which we've always tried to understand network
03:17:00 traffic.

03:17:01 Q. Is that just the way networks work?

03:17:03 A. Yes. The -- the data that's actually being transmitted
03:17:07 typically makes no sense to a monitor, so the only way you
03:17:09 can understand that data is to analyze the headers.

03:17:12 Q. In the period leading up to the patents in this case in
03:17:16 1999, how was the Internet evolving?

03:17:20 A. Fast. It was a wild west. It was a ton of fun. There
03:17:29 was companies springing up right and left to do all kinds
03:17:32 of amazing things with the Internet. People were
03:17:35 contemplating transmitting types of data that, again, today
03:17:39 is very common. But back then, we had never heard of -- of
03:17:42 doing such things. So it was a time of tremendous
03:17:45 activity.

03:17:46 Q. And were you developing operating systems in that time
03:17:49 frame?

03:17:49 A. I was.

03:17:49 Q. What impact would that type of evolution or did that
03:17:53 type of evolution have on developers at that time frame --
03:17:57 in that time frame?

03:17:57 A. Well, most of us had to put a stake in the ground and
03:18:01 say this is where we think the technology is going to go
03:18:06 and try and support that in our operating system.

03:18:08 So in my case, I was interested in live audio and

03:18:12 video. I was very interested in videoconferencing, which
03:18:16 was a very challenging application in the mid to late '90s.
03:18:20 And so I was banking that that would be a thing and was
03:18:24 designing operating systems to support that type of -- of
03:18:28 application. So you had to, as I say, put a stake in the
03:18:32 ground and march towards it.

03:18:35 Q. Was that project you were working on that you were
03:18:37 describing, was that a successful project?

03:18:40 A. I think it was. It attracted the attention of -- of
03:18:43 industry. Intel Corporation funded that research, and they
03:18:47 ended up taking some of the protocol work that we developed
03:18:51 for videoconferencing and putting it into a product -- it
03:18:56 was called ProShare -- that they -- that they sold.

03:19:02 Q. Looking back after a period of evolution like that in
03:19:07 the late 1990s to early 2000s, how do you tell the -- the
03:19:12 wheat from the chaff, if you will?

03:19:14 MR. HOSIE: Your Honor, may we approach?

03:19:16 THE COURT: Approach the bench.

03:19:17 (Bench conference.)

03:19:25 MR. HOSIE: Not to be overly procedural, Your
03:19:28 Honor, but he's moving into opinion testimony, and hasn't
03:19:32 tendered the witness as an expert yet.

03:19:35 THE COURT: I was wondering about that. Is that
03:19:37 your intention?

03:19:37 MR. BURESH: I can certainly do it right now if

03:19:39 that would make the Plaintiff more comfortable.

03:19:41 MR. HOSIE: Thank you.

03:19:41 THE COURT: Well, it needs to be done or else
03:19:44 Plaintiff's first question on cross is, has the Court
03:19:49 recognized you as an expert witness?

03:19:52 MR. BURESH: I agree with that.

03:19:53 MR. HOSIE: Thank you, Your Honor.

03:19:54 THE COURT: Let's proceed.

03:19:56 (Bench conference concluded.)

03:19:58 THE COURT: Let's proceed.

03:20:00 Q. (By Mr. Buresh) And, Dr. Jeffay, we've talked about
03:20:12 your education and your experiences. How -- how does the
03:20:16 education you've had and the experiences you've developed
03:20:19 inform your testimony here today?

03:20:21 A. Well, I think the training that I've had, doing a
03:20:26 dissertation in operating systems and having done research
03:20:31 in the field for -- by the time of these patents, I had
03:20:35 been an active researcher for over 10 years, having worked
03:20:38 with companies, having taught the material to 10 years of
03:20:45 students by then, I think that gives me a good perspective
03:20:48 on the technology and qualifies me to offer the opinions
03:20:51 that I'm going to offer today.

03:20:52 MR. BURESH: Your Honor, I would now tender
03:20:55 Dr. Jeffay as an expert in network monitoring technology
03:20:58 and networking environments.

03:20:59 THE COURT: Is there objection?
03:21:00 MR. HOSIE: There is no objection, Your Honor.
03:21:02 Thank you.
03:21:02 THE COURT: All right. Then without objection,
03:21:04 the Court will recognize this witness as an expert in those
03:21:07 designated fields.
03:21:08 Let's continue.
03:21:09 Q. (By Mr. Buresh) And, Dr. Jeffay, as a developer that
03:21:13 was working in the late '90s into the 2000 time frame and
03:21:17 working on projects in that time frame, looking back, how
03:21:20 do you tell the -- the wheat from the chaff, as far as
03:21:24 ideas go?
03:21:25 A. Well, the ideas are still here today. That's the
03:21:29 wheat. And the ones that fell by the wayside that weren't
03:21:34 adopted for one reason or another, that's unfortunately the
03:21:37 chaff.
03:21:38 Q. Looking back, can you think of examples in the
03:21:43 technology space, I'm not talking about this case, but just
03:21:45 in the technology space of people that have made wrong
03:21:48 design choices?
03:21:49 A. Oh, sure, sure. I mean, in -- in the '90s, I had many
03:21:56 colleagues in academia who left academia to do start-ups.
03:22:02 Everybody -- a lot of people wanted to do start-ups. And
03:22:05 so I had lots of personal friends that were leaving big
03:22:09 companies and doing start-ups and, as I say, colleagues at

03:22:14 other universities who were leaving universities to do
03:22:16 start-ups. And they were tremendously smart people. They
03:22:20 had, from my perspective, great ideas, really exciting
03:22:22 stuff. But the vast majority of them didn't make it for,
03:22:27 you know, one -- one reason or another. Their ideas
03:22:30 just -- just didn't -- they're great ideas, they just
03:22:33 didn't take off.
03:22:34 Q. What about big companies, do big companies sometimes
03:22:38 make the wrong choices?
03:22:40 A. Oh, yeah.
03:22:41 MR. HOSIE: Your Honor, may we approach?
03:22:42 THE COURT: Approach the bench.
03:22:43 MR. HOSIE: Thank you.
03:22:44 (Bench conference.)
03:22:54 MR. HOSIE: Your Honor.
03:22:54 THE COURT: What's your issue, counsel?
03:22:56 MR. HOSIE: Thank you. Your Honor, there's
03:22:56 nothing in this expert's expert report about wheat versus
03:23:00 chaff or who survives and who not or how or why.
03:23:06 THE COURT: So you're objecting that that kind of
03:23:08 questioning is outside the scope of his expert report?
03:23:10 MR. HOSIE: Exactly that, Your Honor.
03:23:11 THE COURT: As long as analogies are used to
03:23:15 elicit the substance of the report, that's not a problem.
03:23:17 If substantively the inquiry goes beyond the scope

03:23:22 of his report, then that's a legitimate objection. I take
03:23:25 it that that's not your objection.

03:23:27 MR. HOSIE: No, but it's -- it's -- I wanted to
03:23:32 flag the issue, Your Honor. He's getting very close.

03:23:34 THE COURT: I'm going to overrule the objection.

03:23:36 MR. HOSIE: Thank you, Your Honor.

03:23:37 (Bench conference concluded.)

03:23:37 THE COURT: Let's proceed.

03:23:48 Q. (By Mr. Buresh) Do you remember the question, or do
03:23:50 you want me to ask it again?

03:23:51 A. Please.

03:23:52 Q. Can you think of a larger company who's made a bad
03:23:55 choice?

03:23:58 A. I don't think -- large companies aren't here anymore.
03:24:03 I hate to say made a bad choice because I understand the
03:24:03 choices they made. They made a choice that didn't work
03:24:07 out.

03:24:08 The one that most comes to mind was a company that
03:24:10 was called Digital Equipment Corporation, which is
03:24:13 shortened as DEC. And throughout the late '70s and
03:24:17 certainly into the '80s and the early '90s, they were a
03:24:23 huge company. They were the No. 2 computer company in the
03:24:25 country. And those of us sort of in the research field
03:24:29 were thinking and hoping they might become the No. 1
03:24:33 company because I used their technology, I bought their

03:24:36 products, and I thought they were great.
03:24:39 They made some bad bets. They didn't really
03:24:43 believe in personal computing, which sounds kind of silly,
03:24:47 but, you know, it was a different world back then. And, as
03:24:49 I say, the future just wasn't clear. And this -- this huge
03:24:53 company is no longer -- it was the No. 2 computer
03:24:57 company -- is no longer with us today.
03:24:58 Q. Do you know whether Digital Equipment had any patents?
03:25:03 MR. HOSIE: Your Honor, may we approach?
03:25:05 MR. BURESH: I'll withdraw the question. It's
03:25:07 fine, Your Honor.
03:25:07 THE COURT: Then having withdrawn the question,
03:25:09 let's move on.
03:25:11 Q. (By Mr. Buresh) Now, all told, you've been a professor
03:25:14 in the computer networking area for how long?
03:25:18 A. Operating system -- in the area of operating systems
03:25:21 and networking for over 30 years.
03:25:24 Q. In addition to teaching, you -- you obviously do some
03:25:27 consulting work; is that fair?
03:25:30 A. I do.
03:25:30 Q. And how often do you do consulting work of -- of this
03:25:33 type?
03:25:33 A. Not terribly often. It's -- I think it officially
03:25:38 consumes about 15 to 20 percent of my time on average.
03:25:42 Q. When was the last time you testified in a setting like

03:25:49 this?

03:25:49 A. In a courtroom like this was 2016.

03:25:54 Q. Why don't you do more of this type of work?

03:25:57 A. Well, you know, it's not my -- I'm a full-time
03:26:03 academic. I mean, I -- I enjoy working on these projects.
03:26:07 They're certainly financially rewarding, but I have a
03:26:11 full-time job, and I love teaching, I love working at the
03:26:14 university, and that's what I want to do full-time.

03:26:18 Q. Before working on this case, had you ever worked with
03:26:23 NetScout before?

03:26:23 A. No.

03:26:26 Q. How about me or my firm?

03:26:29 A. No.

03:26:29 Q. How about Mr. Gillam or his firm?

03:26:33 A. No, I have not.

03:26:34 Q. Now, in the late 1990s time frame, were you familiar
03:26:43 with companies that were working in the operating system
03:26:46 space?

03:26:47 A. Yes. As I say, there -- I had several classmates from
03:26:51 graduate school who went and did start-ups, in particular
03:26:55 in the operating system space.

03:26:57 Q. Had you ever heard of Mr. Balassanian before this case?

03:27:00 A. No, I -- I don't believe I have.

03:27:03 Q. How about BeComm or Implicit?

03:27:05 A. I'm sorry, how about?

03:27:07 Q. How about BeComm or Implicit?
03:27:09 A. No, I don't believe I've heard of them.
03:27:12 Q. Were you familiar with products called Portal or
03:27:18 Strings?
03:27:18 A. No.
03:27:19 Q. Now, as a consultant, are you paid for your time?
03:27:21 A. Yes, I am.
03:27:22 Q. Including while you're here today?
03:27:24 A. Correct.
03:27:25 Q. Does that fact impact your approach or the analysis you
03:27:30 did in this case?
03:27:31 A. No, I -- I believe I am -- I strive to be an
03:27:35 independent expert.
03:27:40 MR. BURESH: If we can go to the next slide,
03:27:42 please.
03:27:42 Q. (By Mr. Buresh) And do you understand that in these
03:27:46 cases, the analysis you conduct is from the perspective of
03:27:51 a person of ordinary skill in the art?
03:27:52 A. I -- I do understand that.
03:27:54 Q. What is a person of ordinary skill in the art?
03:27:57 A. So it's a con -- as I understand it, it's a concept in
03:28:01 patent law. It's the person to whom a patent is directed.
03:28:04 It's the audience of a patent. And on this slide, I've
03:28:09 offered my opinion as to the background of a person of
03:28:13 ordinary skill in the art for the patents in this case.

03:28:19 So, my opinion, they would have a Master's degree
03:28:19 in computer science or a related field, and that they
03:28:22 should have some professional experience working in
03:28:25 computer networking.
03:28:26 Q. What time frame do you assess the person of ordinary
03:28:30 skill in the art?
03:28:30 A. My understanding is that you assess it at the time the
03:28:33 invention was made or conceived, and that, in this case,
03:28:37 the date that we're working with is 1999.
03:28:40 Q. Were you a person of ordinary skill in the art in 1999?
03:28:42 A. Yes. I had at least a Master's degree. I had a Ph.D.
03:28:47 for 10 years by this point. And I have been working for
03:28:51 more than two years in the field of computer networking.
03:28:54 Q. And did you perform your analysis that you conducted in
03:28:57 this case from that perspective?
03:28:59 A. Yes, from the perspective of a person of ordinary skill
03:29:02 in the art.
03:29:02 Q. Could you explain to the jury at a high level the
03:29:08 analysis that you undertook in this case?
03:29:10 A. Sure. So my job was to determine whether or not there
03:29:18 was infringement, and I started with a deep study of the
03:29:21 patents. And we've heard about the proceedings that go on
03:29:25 back and forth between the inventor and the Patent Office.
03:29:27 I was -- it was called the prosecution history. I studied
03:29:29 that.

03:29:30 I read deposition testimony of all of the
03:29:34 witnesses, I believe, that have appeared here, as well as
03:29:39 some other individuals. I reviewed a variety of NetScout
03:29:43 documents. And then the primary thing I did was study the
03:29:48 source code for the accused products.
03:29:50 Q. And how do you make a determination as to whether a
03:29:54 product infringes a patent?
03:29:55 A. Well, we've heard about the claims. So the claims of a
03:29:59 patent are, as I understand it, are what defines the
03:30:04 invention and what you're -- the exercise is to take the
03:30:07 claims and try and match them up against the products and
03:30:10 see if the products have the claims of the patent.
03:30:12 Q. Do you have -- do you have a notebook in front of you?
03:30:21 Behind you, I'm sorry.
03:30:30 A. I have notebooks, but I don't think I'm on them.
03:30:33 Q. Give me just one second.
03:30:53 MR. BURESH: May I approach, Your Honor?
03:30:53 THE COURT: You may.
03:30:54 THE WITNESS: Thank you, sir.
03:30:55 Q. (By Mr. Buresh) You should find a patent -- one of the
03:31:07 asserted patents in your notebook.
03:31:08 A. Yes, I have it here.
03:31:09 Q. Can you pull that out and walk the jury through what
03:31:15 the patent looks like and the parts of the patent and where
03:31:17 they will find the claims?

03:31:19 A. Sure. So I think earlier, you were shown the fancy
03:31:23 version that they call the ribbon copy. This is the style
03:31:29 document that I typically work with.

03:31:31 The front page is -- has some administrative
03:31:33 details about the patent and the dates. There's some
03:31:40 references that are on the next couple of pages.

03:31:44 And then we come to the figures. In this case,
03:31:47 there are 16 -- 16 different figures. And then after the
03:31:51 figures, you have text in a set of numbered columns,
03:31:59 starting with Column 1, and this is -- I think you've heard
03:32:02 the term specification.

03:32:03 This is the specification. This is where the
03:32:06 inventor describes the problem they're trying to solve and
03:32:08 the background, and they describe their invention.

03:32:12 And then at the very back -- in this case, this is
03:32:18 starting at Column 14, so I'm looking at the patent whose
03:32:21 last three numbers end in '683. There's a tradition,
03:32:26 convention of referring to patents by the last three
03:32:30 numbers.

03:32:30 So in Column 14 of the '683 patent, there are a
03:32:34 bunch of numbered paragraphs, essentially. And those are
03:32:36 the claims. And so each one of them, I understand, is an
03:32:40 invention. And the invention is as described in the claim.

03:32:45 Q. Dr. Jeffay, could you show the jury where the
03:32:50 background section of the patent is?

03:32:51 A. Sure. I was just there. So, as I said, the -- the
03:32:58 text is called the specification. And so in Column 1, the
03:33:04 first full section is labeled background. And, as I say,
03:33:07 this is typically where the inventor will describe the
03:33:11 state of the art at the time they filed their patent and
03:33:14 describe the problem that they're trying to solve.

03:33:16 Q. Okay.

03:33:23 MR. BURESH: Mr. Palisoul, can you bring up the
03:33:25 '683 patent at Column 1, Lines 54 to 61, please?

03:33:33 Q. (By Mr. Buresh) Is this a portion of the background
03:33:35 section of the '683 patent?

03:33:35 A. Yes, this is from -- near the bottom of Column 1.

03:33:38 Q. And in the expanded portion here, what is being
03:33:44 described in the background of this patent?

03:33:45 A. So this is the inventor starting to describe the
03:33:51 problem that they see with the -- with the state of the art
03:33:55 as it existed at that time. And what they're specifically
03:33:58 saying -- we can read the text here -- is that at that
03:34:03 time, a computer could receive data in many different
03:34:06 formats, and the format may not be known until the data is
03:34:10 received.

03:34:10 And then the next sentence is the interesting one.
03:34:13 It says that the overhead of statically providing each
03:34:18 possible series of conversion routines is -- is very high.
03:34:21 So they're sort of identifying the problem that they're

03:34:23 going to be concerned with.

03:34:26 Q. How common was it at the time of the '683 patent in 1999 to statically provide each possible series of conversion routines?

03:34:36 A. Well, generally, statically providing function, be it conversion routines or other things, is -- it's the common way programs are written. It's -- it's sort of -- sort of a default that you -- you think about all the cases you want to handle in your program, and you statically develop the program to handle those cases.

03:34:59 Q. How common is it today to statically provide each possible series of conversion routines?

03:35:05 A. I'm sorry, what was the first part of the question? I missed it.

03:35:08 Q. How common is it today to provide -- or statically provide each possible series of conversion routines in a system?

03:35:16 A. In -- in my experience, it's -- it's always been the most common way. It was -- it was common -- there's certainly issues with it, but it's common. It was common in the late '90s, and it's common today.

03:35:28 Q. With reference to the patent, what was the patent describing as the proposed solution to this problem?

03:35:35 A. Well, the proposed solution is -- is described a little further -- further down. But the proposed solution was, in

03:35:44 essence, to dynamically determine the conversion routines
03:35:49 that you're going to use to process data -- newly arrived
03:35:52 data.

03:35:52 Q. What does it mean here that the overhead of the static
03:35:59 approach is very high?

03:36:00 A. Overhead is -- is referring to some measure of cost.
03:36:04 And, for example, the cost of providing routines for every
03:36:08 possible data format is potentially going to be expensive.
03:36:11 You've got to write them all. They have to all be in
03:36:15 memory. So dealing with all possible combinations and
03:36:20 probabilities, it can be expensive.

03:36:22 Q. What is a series of routines, Dr. Jeffay?

03:36:25 A. So that's a phrase I'm sure we've heard a lot. A
03:36:32 routine is -- you can think of it as a function. It's a
03:36:35 processing step that a computer is going to do, typically
03:36:38 made up of many instructions, dozens of instructions. And
03:36:38 a series of routines is basically you do Step 1, then you
03:36:43 do Step 2, then you do Step 3. And a routine is
03:36:47 responsible for doing each one of those steps. So the
03:36:50 sequence of routines is the routines that you're going to
03:36:52 execute to solve some problem.

03:36:57 MR. BURESH: If we could pull up the next slide,
03:36:59 please.

03:36:59 Q. (By Mr. Buresh) What are we seeing here, Dr. Jeffay?

03:37:03 A. So this is now the back of the '683 patent, in

03:37:08 Column 14 that I just showed the jury, and what's
03:37:12 highlighted here is the text for Claim 1.
03:37:15 Q. Could you walk us through the -- the basic concepts in
03:37:20 this claim?
03:37:21 A. Sure. So we can see that the -- the claim is an
03:37:28 apparatus. So they're claiming a device, and it's a device
03:37:31 that's going to include a processing unit, a processor, and
03:37:36 then a memory. And a memory is going to have instructions
03:37:41 that are going to do the following three things: The
03:37:43 create the process, the store, and the process.
03:37:45 And what I've highlighted here is the first step
03:37:48 that the instructions are going to -- that the executing
03:37:52 instructions are going to do. And they're going to
03:37:54 create -- based on information in a received packet,
03:37:58 they're going to create a path that includes a data
03:38:02 structure that indicates a sequence of routines for
03:38:06 processing packets.
03:38:07 Q. Okay. And focusing in on this sequence of routines --
03:38:14 strike that.
03:38:15 Where do you see in this claim the concept of
03:38:18 creating after receiving a packet?
03:38:21 A. So -- so that's not in this language, per se. That
03:38:26 comes from the Court's so-called claim construction. It's
03:38:31 a definition of terms that the Court has provided for us.
03:38:34 MR. BURESH: If we could advance to the next

03:38:36 slide, please.

03:38:37 Q. (By Mr. Buresh) And just to provide a little context,
03:38:43 what does it mean when the Court defines a term of a claim?

03:38:47 A. Well, I can just tell you how I understand it, that
03:38:50 there's often disputes about what terms mean. And so the
03:38:55 Court is asked to define a term. And in this case the
03:38:59 Court has defined for us the phrase "sequence of routines,"
03:39:04 and they've told us that a sequence of routines means an
03:39:11 ordered arrangement of software routines that was not
03:39:15 selected from a set of arrangements created before
03:39:17 receiving a first packet of a message.

03:39:21 Q. And, Dr. Jeffay, did you apply the Court's construction
03:39:24 in your analysis?

03:39:25 A. Yes, I did.

03:39:29 MR. BURESH: If we could go to the next slide,
03:39:32 please.

03:39:32 Q. (By Mr. Buresh) What is depicted here, Dr. Jeffay?

03:39:36 A. So this is a -- a simple diagram to try and illustrate
03:39:39 what I think are some of the core concepts of the asserted
03:39:45 patents. And what I'm showing here is a timeline of
03:39:48 processing of a packet. So time goes to the right here.

03:39:52 And there's three basic points on this timeline.
03:39:56 The first one is that a packet arrives. So a packet is
03:39:59 going to arrive in the system. And then what's going to
03:40:02 happen, according to the claims, is that we do not process

03:40:06 the packet. We -- so that's why I have a stop sign. We
03:40:10 sort of stop. And we create this sequence of routines
03:40:14 that's going to be used to process the packet.
03:40:17 So we stop, we create the sequence of routines,
03:40:20 and then we go ahead and process the packet using the
03:40:23 sequence of routines that we just created.
03:40:26 Q. Do you recall the Lego animation from the opening
03:40:31 statement and Dr. Almeroth's testimony?
03:40:32 A. I do.
03:40:32 Q. How does that align with this timeline you've depicted
03:40:37 here?
03:40:37 A. I think it aligns well. I thought that was a good --
03:40:41 good animation. As I recall, they showed some bricks, a
03:40:45 particular stack, different colors, coming in, stopping,
03:40:48 and then while they stopped, they rummaged around in the
03:40:51 bucket full of Lego pieces to find Lego pieces that matched
03:40:58 the colors on their bricks. And then they snapped them
03:40:58 together.
03:41:01 So then that was -- they have their sequence of
03:41:03 routines. And then the packet continued. And then it was
03:41:05 processed by that particular matching stack of Lego bricks.
03:41:09 So I think it was a -- a -- it was a good way to
03:41:12 represent essentially what I'm trying to represent here.
03:41:15 Q. In the patent, when is the sequence of routines created
03:41:18 in relation to when the packet is received?

03:41:19 A. So -- so this is the important part is that in the
03:41:23 patent, the sequence of routines is created after the
03:41:28 packet arrives and before the packet is processed.
03:41:32 Q. When does the processing of the packet occur in
03:41:35 relation to when the path is created?
03:41:37 A. So the purpose of creating the path is to have a
03:41:42 sequence of routines to process the packet. So you can't
03:41:45 process the packet until you have the sequence of routines.
03:41:48 So processing of the packet will occur after
03:41:53 you've created the sequence of routines.
03:41:54 Q. In the patents, where does the concept of flexibility
03:42:00 come in?
03:42:00 A. It's described in the specification because the idea is
03:42:09 that you have a set of conversion routines to process, and
03:42:14 you can look at a packet and dynamically figure out which
03:42:17 routines you need to use to process that packet so you can
03:42:22 dynamically go in your Lego bucket and find the right
03:42:25 pieces you need and snap them together and make a sequence
03:42:29 of routines to process that packet. So that's -- that's
03:42:32 the flexibility. You didn't have to have those Lego pieces
03:42:35 pre-snapped together before the packet arrived.
03:42:38 Q. Now, did you put together a demonstrative that
03:42:44 illustrates how Mr. Balassanian's patents describe the
03:42:50 operation of a dynamic system?
03:42:51 A. I did.

03:42:52 MR. BURESH: And, Mr. Palisoul, can you first pull
03:42:55 up Figure 3 of the '683 patent?
03:42:57 Q. (By Mr. Buresh) Can you describe that process starting
03:43:01 with this Figure 3?
03:43:02 A. Sure. So this is Figure 3 from the patent, and
03:43:06 remember that Claim 1 of the '683 patent talked about an
03:43:11 apparatus that had a processor and memory. So this is
03:43:15 showing the memory of the invention. And it shows a lot of
03:43:18 components in a memory.
03:43:19 And one of the ones that I want to highlight here
03:43:23 is the one that's outlined in red. And that's my
03:43:25 outlining. The patent didn't outline that.
03:43:27 And so what it is showing is that you have a bunch
03:43:30 of conversion routines in the memory. So these are
03:43:36 effectively the Lego -- the Lego bricks. So you have a
03:43:39 collection of these Lego bricks sitting in memory waiting
03:43:41 to be used.
03:43:42 Q. And what might that look like inside this conversion
03:43:46 routine box?
03:43:46 MR. BURESH: If we can advance the slide.
03:43:49 A. So this is my depiction of what's described in the
03:43:51 patent. But the idea is that you would have routines for a
03:43:57 variety of protocols, and all these protocols here that I'm
03:43:59 illustrating, these are just totally made-up protocols.
03:44:03 There's no such thing called a GG or a PP protocol.

03:44:03 But the idea is that you would have one or more
03:44:11 conversion routines to help you process a packet, and each
03:44:11 conversion routine takes a particular type of input and
03:44:14 generates a particular type of output. And so I sort of
03:44:17 color-coded the inputs, which are on the left of each box,
03:44:22 and the outputs which are on the right. And you sort of
03:44:25 try and hook up compatible inputs to compatible outputs,
03:44:31 meaning, in this case, they would be compatible colors.
03:44:33 So these are -- these are the Lego bricks, and you
03:44:36 will -- will search around in here when a packet arrives --
03:44:39 will search around in here to find a first routine and a
03:44:43 second routine, and then we'll --
03:44:45 Q. Can I stop you there, Dr. Jeffay?
03:44:47 A. Oh, sure, sorry.
03:44:48 Q. Where in here is the processing path?
03:44:50 A. The processing path doesn't exist. So -- so, as I say,
03:44:53 this is the bucket full of Lego bricks. Just -- these are
03:44:58 the building blocks, you know, that are sitting there
03:45:00 waiting to be used.
03:45:01 Q. Okay.
03:45:02 MR. BURESH: And Mr. Palisoul, if we could pull up
03:45:04 the next demonstrative.
03:45:05 Q. (By Mr. Buresh) What happens now when a packet is
03:45:08 actually received in this -- the system described in the
03:45:11 patents?

03:45:11 A. So the patent describes -- describes the invention as
03:45:16 running on end systems, so systems that generate and send
03:45:20 and receive packets.
03:45:21 And in this case, when a system, the one on the
03:45:24 right, receives a packet, what the patent describes is that
03:45:28 you'll -- you'll search through the conversion routines to
03:45:33 try and identify conversion routines to process the packet
03:45:36 according, for example, to the protocols in there.
03:45:38 So this is the process of digging around in the
03:45:40 Lego bucket to find the right color of bricks. So we found
03:45:45 the first routine for protocol PP, we search some more, we
03:45:49 find another conversion routine that is relevant to our
03:45:53 packet. And, in this case, I'm just going to show that
03:45:55 there are four conversion routines that are going to be
03:45:58 necessary to process the packet. So we've identified four
03:46:01 routines.
03:46:01 Q. And when is this processing happening?
03:46:04 A. So this is after the first -- after the first packet of
03:46:07 a message arrives.
03:46:08 Q. Do we have our path at this point?
03:46:10 A. No. So we can't process the packet at this point
03:46:13 because we don't have the sequence of routines to process
03:46:16 it.
03:46:16 Q. So what happens next?
03:46:17 A. So now we're going to glue them together.

03:46:22 Mr. Balassanian described this as sort of a --
03:46:24 building an assembly line, which I thought was a good
03:46:27 example, and so this is sort of the assembly line. And we
03:46:29 are connecting the components of the assembly line
03:46:32 together, in this case using these yellow lines. And you
03:46:35 kind of see that -- notice that the colors of the left side
03:46:39 of the box match up with the colors of the right side of
03:46:42 the box, indicating that these are compatible connections.

03:46:48 Q. At what point then do you have the path?

03:46:48 A. So at this point, we have our sequence of routines, and
03:46:51 we encode this in a data structure, and that would be our
03:46:55 path.

03:46:56 Q. Is this result that you've depicted here consistent to
03:47:01 what is depicted in the patents?

03:47:03 A. Yes. I just made a -- sort of a more fulsome version
03:47:08 of the figure that's in the patent.

03:47:09 So Figure 2, I was essentially illustrating what's
03:47:12 Figure 2, just doing it in a little more colorful --
03:47:17 colorful way. Figure 2 shows the lines being conversion
03:47:21 routines and them being connected together in a path.

03:47:23 Q. And what is happening to the packet while this
03:47:27 processing is going on?

03:47:28 A. Effectively, it's just sitting there waiting to be
03:47:31 processed. We can't process it yet because we don't have
03:47:35 the sequence of routines to process it.

03:47:37 Q. Now, are there more detailed steps for how the sequence
03:47:45 of routines are created than what you've just described?
03:47:49 A. Oh, yes, there's many more steps.
03:47:52 Q. Can you give the jury just a sense -- and I don't want
03:47:55 you to spend a ton of time on this -- but a sense of the
03:48:00 process that's actually described in Mr. Balassanian's
03:48:02 patents?
03:48:02 A. Sure. So, as I mentioned, there are 16 pages of
03:48:06 figures. And the bulk of them describe this process of
03:48:10 searching for routines, finding the right routines, and
03:48:14 connecting them together. And it's done through a variety
03:48:16 of flowcharts.
03:48:18 So here's Figure 1. And if we start at the very
03:48:22 top, there's a box labeled 101 that's called driver. This
03:48:27 is described in the patent as representing the receipt of
03:48:30 the packet. It's a network driver, receives the packet.
03:48:35 If we could just go back to Figure 1.
03:48:36 And then the next -- so we're sort of here. This
03:48:43 is receiving the packet. And then these remaining boxes
03:48:46 here are going to be all involved in the process of trying
03:48:49 to understand the protocols, the formats of the arriving
03:48:54 packet, and the conversion routines that we're going to
03:48:57 need to use to process this packet.
03:49:00 So we go off and we find the conversion routines,
03:49:03 and then we stick them together. And that's effectively

03:49:06 what's described through a number of flowcharts.
03:49:11 So there was a message send box on the previous
03:49:15 slide. This message send here, and that's described in
03:49:18 Figure 7.
03:49:20 So here's a set of steps, a flow chart. It turns
03:49:23 out that it doesn't fit on one page, so this connects here.
03:49:28 So this little bit of a flowchart sticks on over here. And
03:49:30 this flowchart sticks here. So it's kind of a big process.
03:49:35 And these are the steps in the message send box.
03:49:39 Then you have the demux and the label map and our
03:49:43 flowcharts describing this.
03:49:45 This is the flowchart for the demux, which stands
03:49:48 for demultiplex, which is what you do when you receive
03:49:50 something. And it goes through a number of steps, and I
03:49:55 won't go in detail --
03:49:55 Q. Slow down just a little bit.
03:49:57 A. Sorry, I'm going too fast.
03:49:59 Q. Thank you.
03:49:59 A. Apologies.
03:50:00 Some of the boxes here themselves blow up into
03:50:07 additional flowcharts. So this is the -- the get next
03:50:14 binding flowchart that patches into the previous one.
03:50:18 And then if we can go forward, this is -- so we
03:50:22 were going to look at the get key, the get session, and
03:50:25 then the nail binding.

03:50:27 And if we can go to Figure 12 through 14, these
03:50:31 are the flowcharts to get the key, get the session. And
03:50:34 nail the bind -- nail the binding means you're sort of
03:50:40 snapping the pieces together.

03:50:41 So this is -- this is what the patent shows about
03:50:43 the process of creating and -- creating the path.

03:50:48 Q. How complicated is the process of creating the path as
03:50:51 described in the asserted patents?

03:50:52 A. Well, it's hard to characterize in the abstract. The
03:50:56 point is, it's a process. I mean, there's lots of steps
03:50:59 that spread out over five, six pages of flowcharts.

03:51:03 Q. Do all those steps take time?

03:51:05 A. Yes, every step will take some time to -- to execute.

03:51:10 Q. What is happening to the packet and the packets behind
03:51:16 it while that processing is occurring?

03:51:18 A. Nothing can happen because, again, you can't process a
03:51:22 packet until you've built this path, this sequence of
03:51:25 routines. So while you're building the sequence of
03:51:28 routines, everything is going to be on hold.

03:51:30 Q. If you had a system that was processing 8 million
03:51:33 packets per second or 80 million packets per second, would
03:51:37 you have time to do the work that's required to create this
03:51:43 path that we've just walked through?

03:51:44 A. Well, it's hard to answer in the absolute because it
03:51:48 depends on how fulsome you want to do this.

03:51:51 But I think the essential point is, if you're in
03:51:56 the 80 -- 8 million packets or 80 million packets per
03:51:59 second world, you've got frightfully little time between
03:52:02 packets. You have to patent -- the coin of the realm is
03:52:05 you process the packet before the next -- for monitoring
03:52:08 systems, you process the packet before you see the next
03:52:11 packet. You just have to do that.
03:52:14 And if you're going at a hundred gigabits or 400
03:52:17 gigabits -- and these speeds are incredible. You're
03:52:17 literally in the billionths of a second range. So you
03:52:22 really do not have much time to do much of anything. And
03:52:23 you're not going to want to be doing dynamic creation of
03:52:27 paths when you're processing network packets at that speed.
03:52:31 Q. So what was your impression of the process described in
03:52:35 Mr. Balassanian's patents?
03:52:37 A. I understood the process, and I can understood -- I
03:52:41 understood, I think, where he was coming from in 1999 and
03:52:45 for the types of end system devices that were described in
03:52:49 the patent. I mean, it made sense as a good -- good thing
03:52:51 to do.
03:52:52 But that's just not the world that we're in today,
03:52:55 and that's not the world where the accused products live.
03:53:02 MR. BURESH: If you can go to the next slide,
03:53:05 please?
03:53:05 Q. (By Mr. Buresh) Dr. Jeffay, how do you design or build

03:53:08 a traditional static packet processing system?
03:53:12 A. So we've heard testimony about this, and I can confirm
03:53:16 that what you've heard exactly fits with my understanding.
03:53:21 When we built our packet processing systems at
03:53:26 UNC, we start with the specification of the protocol that
03:53:28 tells us where -- how you find things that you're
03:53:31 interested in the packet, so you analyze the spec, and you
03:53:34 write routines to process the portions of the packet that
03:53:38 you're interested in.
03:53:39 So what's shown here on this demonstrative is an
03:53:42 example of an ethernet packet that ultimately contains some
03:53:46 HTTP data, and if we were going to write software to
03:53:51 process this statically, we'd look at the specification for
03:53:55 HTTP and TCP and IP and ethernet. And we'd probably build
03:54:01 modules, a module to deal with the ethernet, the first bit;
03:54:05 and then a module to deal with IP, the second bit; and then
03:54:10 a module to deal with TCP; and then finally a module to
03:54:14 deal with HTTP. So you statically write this code, build
03:54:17 these modules, and then you would connect them together at
03:54:21 the time that you write the code.
03:54:22 And so --
03:54:23 Q. And how do you connect them together in this particular
03:54:25 order?
03:54:25 A. So it gets a little technical, but this is traditional
03:54:28 programming, and programming is all about hooking pieces of

03:54:31 code together. So you've seen examples here of this, you 03:54:35 know, if/then/else statements and case and switch 03:54:39 statements. But there's programming language statements 03:54:41 you use to connect these things together, and you get 03:54:43 this -- this hard-wired or, you know, welded together set 03:54:48 of pipes.

03:54:48 Q. Can you describe whether in a static system it's 03:54:53 possible to change the order?

03:54:54 A. You can change the order. You just have to reprogram 03:54:58 the system manually. I mean, the programmer who wrote the 03:55:01 system originally has to go back, and, you know -- so think 03:55:05 of these as literally pieces of PVC pipe that are bonded 03:55:09 together. You have to go in and cut the bonds and stick in 03:55:13 a new piece of pipe and reconnect it.

03:55:15 And so a programmer -- in a static system, a 03:55:18 programmer has to go and write the code and connect it in 03:55:22 wherever they want in this path.

03:55:24 MR. BURESH: If you could advance the slide.

03:55:30 Q. (By Mr. Buresh) Now, depicted on the lower left-hand 03:55:32 is a different packet, correct?

03:55:34 A. Correct.

03:55:35 Q. And what would have to be done if you wanted your 03:55:39 system to be able to process this type of packet in a 03:55:42 static system?

03:55:43 A. So in the static world, we just follow the same process

03:55:46 again. So this is an ethernet packet containing what's
03:55:50 called SMTP data. The M in -- in SMTP stands for mail, so
03:55:57 this is -- this is a protocol that's used to deliver --
03:55:59 deliver your email.

03:56:00 So you analyze this packet, and you say, well,
03:56:02 it's got ethernet, so I need to process ethernet. But the
03:56:04 good news is I have ethernet. I need to process IP, but
03:56:09 the good news is I have IP. I need to process TCP. I have
03:56:13 TCP.

03:56:13 So all -- in this case, all I have to do is write
03:56:17 a new module to do SMTP processing and then connect it in
03:56:21 to the TCP module.

03:56:23 Q. And how do you connect it into the SMTP module?

03:56:28 A. Again, you do standard programming techniques. You
03:56:33 have a function, and you will have something that
03:56:34 ultimately leads to a function call.

03:56:36 Q. Now, is this order 1, 2, 3, 4, is that fixed in a
03:56:40 static system?

03:56:41 A. Yes. For processing ethernet packets containing SMTP
03:56:46 data, they will -- in this static system, they will always
03:56:50 be processed in this order: Ethernet, then IP, then TCP,
03:56:56 then SMTP.

03:56:57 MR. BURESH: If you can advance the slide, please.

03:57:02 Q. (By Mr. Buresh) Depicted on the lower left is a third
03:57:05 type of packet. Do you see that?

03:57:07 A. I do.

03:57:07 Q. And, again, one more time, what would you -- what would
03:57:10 a developer do in a static system if they wanted to process
03:57:13 this type of packet?

03:57:14 A. Okay. We run through the process again. We analyze
03:57:17 the protocols. This is an ethernet packet containing
03:57:20 what's called RTP. RTP is a protocol used to transmit
03:57:20 voice and video.

03:57:25 We recognize that we have ethernet. And we have
03:57:27 an ethernet module. We have IP. We have an IP module. We
03:57:31 have a new Layer 4, L4 header. UDP, we don't have one of
03:57:37 those, so we're going to have to write a module for UDP.
03:57:40 And then we're going to have to write a module for RTP.

03:57:44 And, ultimately, the UDP is going to get hooked in
03:57:48 to IP, and then UDP will get hooked into RTP. And so now
03:57:52 we have another path, another -- another set of pipes in
03:57:55 our -- in our plumbing here.

03:58:05 MR. BURESH: If we could bring up the next slide.

03:58:07 Q. (By Mr. Buresh) Now, I'd like to put the -- the
03:58:13 system, take it out of the development context and put it
03:58:15 into a network, all right?

03:58:17 A. Okay.

03:58:19 Q. In a static system, what will happen when a packet like
03:58:24 the one depicted here is received into the system?

03:58:27 A. So for this processing system, if it receives this

03:58:31 packet, this would be an ethernet packet containing HTTP
03:58:34 data. You simply start processing the packet, going from
03:58:41 the left to the right, processing ethernet, then IP, then
03:58:44 TCP, and then HTTP.

03:58:50 And in this case, we specifically have plumbing to
03:58:50 deal with this. We specifically have a path, so it would
03:58:52 follow the path that's illustrated in green.

03:58:56 Q. What degree of variability is there in any -- in a
03:59:01 static system for processing a packet of this type?

03:59:03 A. Zero. The downside of static is that it's -- if you
03:59:09 need to make changes, you have to re-program it. You
03:59:13 have -- a programmer has to go back, write new code, link
03:59:16 in the code, and then ship it back out again.

03:59:20 Q. How does the -- the fact that this path is being used
03:59:27 to process this packet and not the other paths, how does
03:59:31 that relate to path creation?

03:59:35 A. It doesn't really. I mean, all these paths -- so here
03:59:39 is a static system that has paths that are sufficient for
03:59:42 processing three types of data, web data, email data --
03:59:47 sorry, web -- ethernet packets with web data, ethernet
03:59:51 packets with email, and ethernet packets with video. So it
03:59:57 has three static paths. And the fact that we're using the
03:59:57 one right now for HTTP, the other paths are still there,
04:00:00 we're just not using them for this particular packet.

04:00:03 Q. When were these paths created in a static system?

04:00:07 A. They're all created at the time the system was designed and coded.

04:00:15 MR. BURESH: If we could go to the next demonstrative, please.

04:00:19 Q. (By Mr. Buresh) Do you recall from the opening demonstrative that Mr. Hosie used and the same Lego demonstrative that Dr. Almeroth used, they had a static system depicted, as well? Do you recall that?

04:00:33 A. I do.

04:00:33 Q. And in that system, the paths were already there, right?

04:00:37 A. Correct.

04:00:37 Q. Do you recall a yellow/black, yellow/black packet that came in to their system?

04:00:47 A. Yes. Late, late in their animation, yes.

04:00:50 Q. And then a question mark appeared?

04:00:52 A. Yes.

04:00:52 Q. What happens in a static system when a packet arrives that you don't already have a path created for?

04:00:59 A. So if you don't have the plumbing, you can't process that -- that packet. And you can only process packets for which you have patents.

04:01:13 Q. And, Dr. Jeffay, depicted on Slide 23 here, we have a packet that has ethernet, IP, TCP, and RTP data. Do you see that?

04:01:19 A. I do.
04:01:20 Q. Is there plumbing to process this packet in the example
04:01:24 static system you have depicted here?
04:01:26 A. Well, this one is interesting because from a module
04:01:29 standpoint or from a routine standpoint, we have all the
04:01:33 routines. We have an ethernet routine. We have an IP
04:01:36 routine. We have a TCP routine. And we have an RTP
04:01:40 routine. So it looks like we're good. We should be able
04:01:42 to process this.
04:01:43 But, although we have all the routines, we don't
04:01:46 have them in the right order, and we don't have them
04:01:49 connected together. There's no connection from the TCP
04:01:51 module to the RTP module, so even though we have -- in
04:01:59 theory we have the code, we can't process this.
04:02:01 THE COURT: Let me interrupt at this point.
04:02:03 Ladies and gentlemen, we're going to take a short
04:02:05 recess.
04:02:06 If you will simply leave your notebooks in your
04:02:08 chairs. Follow all the instructions I've given you. Use
04:02:11 this opportunity to stretch your legs and get a drink of
04:02:13 water, and we'll be back in here shortly to continue.
04:02:17 Of course, you'd expect me to remind you not to
04:02:19 discuss the case among each other.
04:02:21 With that, the jury is excused for recess.
04:02:27 COURT SECURITY OFFICER: All rise.

04:02:46 (Jury out.)

04:02:52 THE COURT: Dr. Jeffay, I want you to slow down as
04:02:55 you testify.

04:02:56 THE WITNESS: Thank you, Your Honor.

04:02:56 THE COURT: You are -- you are very fast, and this
04:03:00 late in the day, without you slowing down, there's a good
04:03:04 chance much of what you say will not register with the jury
04:03:07 or the Court. And without that happening, you're really
04:03:14 wasting your time and our time. So I'm going to expect to
04:03:17 you slow down, okay?

04:03:18 THE WITNESS: I appreciate that, Your Honor. I
04:03:21 will do my best.

04:03:22 THE COURT: We stand in recess.

04:17:39 (Recess.)

04:17:41 (Jury out.)

04:17:43 COURT SECURITY OFFICER: All rise.

04:17:44 THE COURT: Be seated, please.

04:17:44 Mr. Buresh, are you prepared to continue with your
04:17:54 direct examination?

04:17:54 MR. BURESH: I am, Your Honor.

04:17:55 THE COURT: If you'll return to the podium.

04:17:57 While he's doing that, if you will bring in the
04:18:00 jury, please.

04:18:01 COURT SECURITY OFFICER: All rise.

04:18:20 (Jury in.)

04:18:22 THE COURT: Please be seated.

04:18:29 All right. Counsel, you may continue with your
04:18:34 direct examination.

04:18:36 MR. BURESH: Thank you, Your Honor.

04:18:37 Q. (By Mr. Buresh) In the static system we were talking
04:18:44 about before the break, do you recall that, Dr. Jeffay?

04:18:49 A. I do.

04:18:49 Q. There's processing paths for three types of packets,
04:18:53 right?

04:18:53 A. In this example, yes.

04:18:55 Q. Now, what if you wanted to have a static system that
04:18:58 could process, let's just say, a thousand different
04:19:02 protocols, what would you have to do?

04:19:03 A. You would just extend the plumbing diagram to add in a
04:19:12 thousand different paths. You would explicitly program in
04:19:17 the other thousand paths.

04:19:19 Q. Why would a packet inspection system developer want to
04:19:26 do it that way?

04:19:27 A. Speed -- hundred percent speed.

04:19:29 Q. What type of products are the NetScout products that
04:19:37 are at issue in this case?

04:19:38 A. There are all types of -- the three products are all
04:19:44 instances of network monitoring devices for slightly
04:19:48 different purposes.

04:19:52 Q. Now, how did you gain your understanding of the

04:19:55 NetScout products?
04:19:57 A. As I think I said, I -- I looked at a variety of
04:20:01 NetScout documents. I read the deposition transcripts of
04:20:07 the witnesses who were deposed who were knowledgeable about
04:20:09 the products, and then I studied the source code for the
04:20:13 products.
04:20:15 Q. Why is it important to study the source code?
04:20:18 A. The source code tells you exactly how the system
04:20:23 operates. It's -- it's the equivalent of a wiring diagram
04:20:27 or -- or the schematics. If you're trying to repair some
04:20:30 appliance, it tells you exactly how -- how it's put
04:20:34 together and how it operates.
04:20:36 Q. And how do the NetScout products -- let me start that
04:20:42 over.
04:20:43 Did you review the source code for each of the
04:20:45 NetScout products at issue in this case?
04:20:46 A. Yes.
04:20:48 Q. And how do the NetScout products operate with respect
04:20:51 to creating a processing path?
04:20:54 A. Well, they don't. They don't create processing paths.
04:21:01 The -- based on my analysis of the code, they're
04:21:04 constructed using very traditional conventional programming
04:21:07 practices where you create the programming routines when
04:21:13 you design the product, and you arrange them however you
04:21:16 want to arrange them to perform whatever function you need,

04:21:20 and they're connected together. And that's all done at the
04:21:23 time the product is -- is designed.
04:21:26 Q. How do you know that the processing paths in the
04:21:29 NetScout products are all created at the time the system is
04:21:33 designed or built?
04:21:35 A. I mean, it's -- it's evidence in the code. I mean,
04:21:39 you've seen the routines, you see how they're called, and
04:21:43 there's no code in the NetScout products to do any dynamic
04:21:49 creation of -- of processing paths.
04:21:52 MR. BURESH: Okay. If we could move to the next
04:21:55 slide, please.
04:21:56 Q. (By Mr. Buresh) Does this slide depict an example of
04:22:01 InfiniStream processing paths?
04:22:04 A. Yes. This is simplified -- very simplified version for
04:22:07 a couple -- a couple of processing paths within the
04:22:11 InfiniStream product.
04:22:13 Q. And how are the paths connected together or created --
04:22:19 let me start over.
04:22:20 How are the paths connected together as depicted
04:22:24 here in the InfiniStream?
04:22:25 A. They're connected together using, as I say,
04:22:29 conventional programming practices. They are programming
04:22:35 languages, statements that are used to go from one
04:22:36 module -- in this case, they're all called parsers -- to go
04:22:39 from one parser to another parser.

04:22:41 And we've already seen examples of these things,
04:22:43 like these function calls and the case statements and the
04:22:48 if/then statements.

04:22:51 Q. As an example, would you be able to show me in the
04:22:55 InfiniStream source code where the TCP parser is connected
04:22:57 to the HTTP parser?

04:22:58 A. Sure.

04:23:05 MR. BURESH: Mr. Palisoul, can you go to split
04:23:07 screen and show DX-453 at 551?

04:23:14 Q. (By Mr. Buresh) What is the jury seeing here?

04:23:15 A. So this is code that I believe was presented to the
04:23:18 jury previously. It's a -- it's a snippet of code from a
04:23:24 file where about 300 -- almost 400 lines into this file.

04:23:28 And this is what's called a switch or a case
04:23:31 statement where it's -- it's essentially enumerating the
04:23:36 junctions in the plumbing back here. And this is a
04:23:41 particular case where you're going from TCP parsing to HTTP
04:23:49 parsing, and the first highlight shows that for the
04:23:52 particular case of an application flow-type HTTP/TCP,
04:23:58 you're going to use an -- a parser function that's called
04:24:01 the URL parser, URL being a -- an HTTP concept.

04:24:05 Q. Are these instructions?

04:24:07 A. Yes, these are instructions.

04:24:08 Q. And what does the InfiniStream do when it receives, for
04:24:13 example, a packet that has both TCP and HTTP headers?

04:24:17 A. This code will be executed.
04:24:19 Q. Every time?
04:24:21 A. Yes.
04:24:22 Q. How much variability is there in the paths in the
04:24:28 InfiniStream?
04:24:29 A. Zero. I mean, the paths are what the paths are, and
04:24:32 they're created when the product is designed, and that's
04:24:36 all the product can do.
04:24:37 Q. And these instructions that we are seeing here to
04:24:42 connect the TCP parser to the HTTP parser, are there
04:24:46 similar instructions for all of the connections in the
04:24:50 paths in the InfiniStream?
04:24:51 A. So I think you just had that a little bit reversed.
04:24:54 But, yes, there are similar functions for all -- similar
04:24:56 code to this at all of the junction points in the full
04:25:01 version of the plumbing diagram.
04:25:03 Q. Are the paths in the InfiniStream hard-coded?
04:25:06 A. Yes.
04:25:15 MR. BURESH: If we can go to the next
04:25:15 demonstrative, please?
04:25:15 Q. (By Mr. Buresh) What is this depicting, Dr. Jeffay?
04:25:18 A. So this is more interesting looking plumbing, but this
04:25:22 is a simplified version of the processing paths that are
04:25:27 part of the GeoProbe product.
04:25:32 Q. Did you hear Mr. Curtin's testimony earlier today?

04:25:35 A. I did.
04:25:36 Q. Was your review of the software in the GeoProbe
04:25:41 products consistent with the testimony you heard?
04:25:43 A. Yes.
04:25:44 Q. Where are the processing paths in the GeoProbe created?
04:25:49 A. As we heard from Mr. Curtin, they're created when the
04:25:54 software is developed by -- initially developed by
04:25:58 NetScout.
04:25:58 Q. How much variability is there in the processing paths
04:26:02 that will be utilized to process packets in the GeoProbe
04:26:06 software?
04:26:06 A. There's none, and this is the downside of the static
04:26:11 processing approach. You can only process according to the
04:26:17 paths that you have.
04:26:18 Q. In the GeoProbe, are there connections between the
04:26:22 modules?
04:26:23 A. Yes.
04:26:24 Q. Are those programs similar to the example we saw from
04:26:27 the InfiniStream a moment ago?
04:26:28 A. Yes, there's examples here where they use the same
04:26:32 exact programming construct, this -- this switch and case
04:26:36 statement.
04:26:37 Q. Are the connections in the GeoProbe hard-coded?
04:26:40 A. Yes.
04:26:43 Q. Does the GeoProbe product have the ability to change

04:26:50 the order of processing after a packet is received?
04:26:54 A. No, it can -- it can only process according to the
04:26:57 fixed paths that were created when the product was
04:27:02 designed.
04:27:02 Q. Now, if the GeoProbe receives a packet that is -- it
04:27:07 does not have programming to support, an unsupported
04:27:11 protocol, what happens to that packet?
04:27:13 A. Well, essentially, they have the ability to -- to
04:27:18 recognize it as an unknown packet, and it will be processed
04:27:22 as -- as an unknown packet, which means that it really
04:27:27 can't do much processing on.
04:27:31 Q. On the InfiniStream, if the InfiniStream receives a
04:27:34 packet that is not supported, doesn't have software to
04:27:38 process it, what happens to the -- that packet?
04:27:41 A. It's conceptually the same. It gets labeled as an
04:27:46 unknown packet, and it cannot be processed in much detail.
04:27:50 MR. BURESH: If we can go to the next slide,
04:27:56 please.
04:27:56 Q. (By Mr. Buresh) What are you depicting here,
04:27:58 Dr. Jeffay?
04:27:58 A. So -- so this is a slide that we've all seen before,
04:28:01 and this is a rendition of the plumbing for the Arbor
04:28:04 product.
04:28:05 Q. Did you hear Dr. Dawson's testimony earlier today?
04:28:10 A. I did.

04:28:10 Q. And did the testimony you heard from him about the
04:28:13 Arbor products, was that consistent with what you saw in
04:28:16 your review of the Arbor source code?
04:28:19 A. Yes.
04:28:20 Q. When are the processing paths created in the Arbor
04:28:23 products?
04:28:26 A. As we -- as we heard from Dr. Dawson and as my review
04:28:30 of the code indicates, the paths are static, they're
04:28:33 created at the time the software was originally authored.
04:28:39 Q. Now, these connections between routines, like the one
04:28:41 I've highlighted here between the IP and TCP module, are
04:28:51 those hard-coded?
04:28:52 A. Yes.
04:28:52 Q. And are the instructions to do so similar to what we
04:28:56 saw with respect to the InfiniStream?
04:28:58 A. Yes. I think this example here also relies on the same
04:29:00 type of programming language structure, this thing called a
04:29:03 switch or a case statement.
04:29:04 Q. Based on your analysis, if the Arbor products receive a
04:29:09 packet that is in an unsupported protocol -- or an
04:29:13 unsupported protocol, what happens to that packet?
04:29:15 A. They're not -- the Arbor product will not be able to do
04:29:21 much processing. We won't be able, for example, to do any
04:29:24 of the applications, specific countermeasures, and that
04:29:28 packet is just going to be forwarded to its destination.

04:29:31 Q. What degree of flexibility or variability is there in
04:29:37 the processing paths in the Arbor products after a packet
04:29:40 is received?

04:29:42 A. There's none.

04:29:44 Q. From an engineering standpoint, were the NetScout
04:29:50 Systems implemented as you would have expected?

04:29:55 A. Yes. Given that they're monitoring products and given
04:29:58 the speeds of networks today, you optimize for speed, and
04:30:03 flexibility is great, but flexibility frequently comes at
04:30:10 the cost of speed.

04:30:11 And, today, the vast majority of protocols are
04:30:14 known. We've -- we've heard that these devices cover
04:30:18 thousands of protocols, so it makes sense to program
04:30:22 patents or hard-code them in. And this way at run time
04:30:26 when the system is out in the field, it will process
04:30:28 packets as fast as it possibly can.

04:30:30 Q. In your view of the source code for these three
04:30:35 products, the Arbor, the InfiniStream, and the GeoProbe,
04:30:38 did you find any evidence of a processing path being
04:30:41 created after receiving a first packet of the message?

04:30:45 A. I did not.

04:30:46 Q. What type of paths does NetScout use?

04:30:52 A. Again, they're static paths. They're -- they're
04:30:56 hard-coded, pre-created, and they all are existing prior to
04:31:01 the receipt of any -- of any packet.

04:31:04 MR. BURESH: Mr. Palisoul, could you bring up
04:31:08 Claims 1 and 10 of the '683 patent, Claim 1 of the '790
04:31:14 patent, and Claim 1 of the '104 patent on the same screen,
04:31:24 please? Thank you.
04:31:24 Q. (By Mr. Buresh) Now, Dr. Jeffay, are you familiar with
04:31:27 the claim term sequence of routines?
04:31:28 A. I am.
04:31:29 Q. And does that appear in every claim asserted in this
04:31:31 case?
04:31:31 A. It does. For Claim 10 of the '683 patent, the language
04:31:37 is a little different. Instead, a sequence, it's a list,
04:31:42 so it's a list of conversion routines.
04:31:44 Q. And were the phrases sequence of routines and list of
04:31:49 conversion routines construed by the Court?
04:31:51 A. Yes, they were.
04:31:53 MR. BURESH: If we could bring up the next slide,
04:31:56 please.
04:31:56 Q. (By Mr. Buresh) And are these the claim constructions
04:31:59 for those phrases that we just saw in the claims?
04:32:02 A. Yes.
04:32:03 Q. Did you apply the Court's claim constructions in your
04:32:07 analysis?
04:32:08 A. I did.
04:32:11 Q. In light of your analysis that you conducted with the
04:32:15 InfiniStream, Arbor, and GeoProbe source code, do you have

04:32:18 an opinion as to whether NetScout infringes any of the
04:32:21 asserted claims in this case?
04:32:22 A. I do. And my opinion is that they do not infringe, for
04:32:30 at least the reason that they do not contain the sequence
04:32:32 of routines or the list of conversion routines that's
04:32:35 required by all of the claims because they do not satisfy
04:32:39 the Court's definition of those phrases.
04:32:44 Q. Were you here for Dr. Almeroth's testimony?
04:32:47 A. I was.
04:32:49 Q. Did you hear him testify that the NetScout products do
04:32:54 infringe?
04:32:54 A. I did.
04:32:55 Q. What do you think of his analysis?
04:32:59 A. Well, obviously, I disagree. I guess what I would say
04:33:05 is it strikes me as he's combining the create -- what he
04:33:12 calls the creation of the paths with what is really the
04:33:15 processing of the packet.
04:33:18 MR. BURESH: If we could go to the next slide,
04:33:21 please.
04:33:21 Q. (By Mr. Buresh) Is this your timeline of the dynamic
04:33:24 system of Mr. Balassanian's patents?
04:33:25 A. Yes.
04:33:27 Q. And where on this timeline is Dr. Almeroth focusing?
04:33:33 A. I believe, given the code that Dr. Almeroth has
04:33:37 referenced, what he's really referring to is the actual

04:33:40 processing of the packet. He's -- he's pointing to
04:33:43 activities that are occurring while the packet is actually
04:33:46 being processed. So he's -- he's -- he's over here.
04:33:56 Q. And in -- in comparison to this dynamic system, what
04:34:01 would a timeline look like in NetScout's system?
04:34:06 A. Well, it would be a very expanded timeline because it
04:34:10 would have to represent a longer period of time. This
04:34:13 timeline, all things considered, is a very short period of
04:34:16 time. You have a packet arriving, creating routines, and
04:34:19 processing the packet.
04:34:20 The -- the NetScout products are, as I've
04:34:24 illustrated here, they were done to the left which
04:34:29 represents longer ago in history, years ago. So over here
04:34:35 we have fractions of a second, but the arrangements, the
04:34:38 paths were created months, years ago when the software was
04:34:42 actually developed.
04:34:43 Q. Do you recall testimony about the creation of an entry
04:34:55 in a flow table?
04:34:57 A. I do.
04:34:58 Q. And what does creating a flow entry in a flow table
04:35:03 have to do with creating a path for processing packets?
04:35:12 A. Nothing.
04:35:12 Q. What is a flow table?
04:35:14 A. So we talked about flows. Flows, remember, are a
04:35:20 series of packets that are related because an application

04:35:23 will send a piece of data that is too big to fit in one
04:35:27 packet, so it gets chopped up into other packets. And that
04:35:32 collection of packets becomes a flow, and a flow table is
04:35:37 used in the -- the accused products to just maintain
04:35:37 statistics about that flow. And we heard testimony about
04:35:41 this. They count the number of packets, they count the
04:35:44 number of bytes represented by those packets.

04:35:47 And these sort of data are useful for network
04:35:51 managers to understand the traffic in their network.

04:35:56 Q. Now, when along -- when, during processing of a packet,
04:36:11 is the flow entry for that packet created?

04:36:17 A. It's certainly created during the processing of that
04:36:24 packet. As I view it, it's created -- the processing is
04:36:30 done well into the processing of -- of that packet. Yeah,
04:36:35 so you've already done significant processing of the packet
04:36:38 by the time you've -- you're ready to update the flow
04:36:41 table.

04:36:42 Q. How far along -- let's take the Arbor product.

04:36:47 How far along the processing path are you before
04:36:52 you even access the flow table?

04:36:54 A. You're pretty far along. In some of their figures,
04:36:59 diagrammatically it looks like, you know, you're on the
04:37:01 order of half to two-thirds of the way through their
04:37:05 flowchart.

04:37:06 Q. What does that tell you?

04:37:07 A. Well, it confirms to me that the operations on the flow
04:37:11 table are really related to the processing of the packet.
04:37:15 I mean, the purpose of the flow table is to -- it's the --
04:37:22 it's the memory that you're using to record statistics
04:37:26 about the packet, and you get those statistics by
04:37:30 processing the packet.

04:37:31 Q. Do you recall any testimony from Dr. Almeroth that
04:37:35 would speak to this issue?

04:37:36 A. I do.

04:37:40 MR. BURESH: If we can pull up yesterday
04:37:43 afternoon's trial transcript, Page 21, Lines -- around 15
04:37:52 to 17.

04:37:56 Q. (By Mr. Buresh) Do you recall this testimony,
04:37:59 Dr. Jeffay, from Dr. Almeroth?

04:38:03 A. Yes.

04:38:06 Q. And what is he saying here?

04:38:08 A. Well, this was in some Q&A that the two of you had
04:38:14 where you were asking him questions about a system that was
04:38:19 static -- a statically configured system. So it -- it
04:38:24 didn't -- all the paths were created ahead of time before
04:38:28 the first packet in the system.

04:38:29 And you were asking him, if you took such a system
04:38:32 and you were putting data in a flow table, if you're
04:38:37 manipulating flow entries, that that system wouldn't
04:38:41 infringe. And he agreed with you. And -- and I would

04:38:43 agree with him.
04:38:44 Q. What does putting a flow entry in the flow table have
04:38:51 to do with whether or not you've created a path before or
04:38:57 after receiving a packet?
04:38:59 A. So I think what this testimony shows is that that's not
04:39:04 the question to ask. It doesn't -- the issue is not the
04:39:06 flow table and the flow entries. I mean, Dr. Almeroth has
04:39:09 admitted you can have flow entries in a static system, and
04:39:16 they don't infringe. And he's also alleged you can have
04:39:16 flow entries in a dynamic -- what he believes is a dynamic
04:39:19 system, and that infringes.
04:39:20 So the flow entries aren't the issue. The issue
04:39:22 is, is the sequence of routines that's processing the
04:39:26 packet, is it created before or after you receive the first
04:39:31 packet of a message.
04:39:41 Q. Which of those options fall within the scope of
04:39:44 Mr. Balassanian's patents?
04:39:46 A. So Mr. Balassanian's patents require that the sequence
04:39:49 of routines is created after you receive the first packet
04:39:54 of the message.
04:39:54 Q. And when are the sequence of routines created in the
04:40:00 NetScout products?
04:40:01 A. They're created, as shown on the slide, long before
04:40:07 packets arrive. They're -- they're created when the system
04:40:09 is designed.

04:40:10 Q. Do the NetScout products create any sequence of
04:40:15 routines after a packet of a message has arrived?
04:40:18 A. No.
04:40:18 Q. And have you reached a conclusion as to whether the
04:40:21 NetScout products infringe the asserted claims?
04:40:23 A. Again, I have reached a conclusion. And it's my
04:40:26 conclusion that they do not infringe the asserted claims
04:40:36 for at least the reason that they don't have the required
04:40:36 sequence of routines.
04:40:39 MR. BURESH: If we can go to Claim 1 of the '683
04:40:43 patent again.
04:40:43 Q. (By Mr. Buresh) And, Dr. Jeffay, if you could look at
04:40:46 the last limitation of the claim.
04:40:47 A. Sure.
04:40:48 Q. What is that limitation about?
04:40:49 A. So just to reorient with Claim 1, so we have our
04:40:54 processing unit, our memory that's storing instructions,
04:40:57 and instructions. We execute them to create the path,
04:41:02 store the path, and then now process additional packets in
04:41:06 the message.
04:41:07 And as part of that processing, the sequence of
04:41:11 routines that was created have to use a Transmission
04:41:16 Control Protocol to convert packets having a TCP format
04:41:20 into a different format.
04:41:21 Q. Do the NetScout products do that?

04:41:26 A. They do not.
04:41:27 Q. Now, at a high level, why not?
04:41:29 A. Well, there's a couple reasons. The first one is
04:41:36 that --
04:41:36 MR. BURESH: If we could advance the slide,
04:41:39 please.
04:41:39 A. -- there are specific constructions for these terms.
04:41:43 The Court has told us what it means to execute a
04:41:48 Transmission Control Protocol. And it specifically means
04:41:50 that you operate on packets whose outermost header is a TCP
04:41:56 header.
04:41:56 And so based on my analysis, and I'm sure we'll go
04:41:59 into this, I believe the accused products never operate on
04:42:03 a packet whose outermost header is a TCP header.
04:42:07 And then, in addition, they also don't do the
04:42:10 required converting from a TCP format into a different
04:42:13 format because they don't convert the outermost header from
04:42:16 TCP to another header structure.
04:42:18 Q. Do the NetScout products ever operate on a packet whose
04:42:31 outermost header is a TCP header?
04:42:32 A. No, they do not.
04:42:33 Q. Now, we've seen a number of TCP routines, even in the
04:42:39 depictions you've used today, right?
04:42:42 A. That is -- that is true. There's lots of -- lots of
04:42:46 TCP routines.

04:42:47 Q. Why don't those routines satisfy this execute a
04:42:53 Transmission Control Protocol claim limitation?
04:42:56 A. Because the issue isn't the existence of a TCP routine.
04:43:00 The issue is, whatever routine you're looking at, what is
04:43:05 that routine operating on?
04:43:07 So the Court's construction says: Operate on one
04:43:10 or more packets whose outermost header is a TCP header.
04:43:15 So you can call a routine anything you want, but
04:43:18 the fundamental issue is, is that routine operating on a
04:43:21 packet whose outer header is a TC -- outermost header is a
04:43:29 TCP header.
04:43:31 Q. When you were reviewing the source code for the
04:43:36 InfiniStream, the GeoProbe, and the Arbor products, did you
04:43:40 see any indication of something called a representation of
04:43:43 a packet?
04:43:46 A. No. All you see in the products are just packets, and,
04:43:52 specifically, ethernet packets.
04:43:53 Q. Now, you've been teaching in this area for 30-plus
04:44:04 years, correct?
04:44:04 A. Yes.
04:44:05 Q. Prior to Dr. Almeroth's testimony, have you ever heard
04:44:10 anyone saying a pointer in a packet creates a
04:44:12 representation of a packet?
04:44:13 A. No. No. A pointer in a packet is just -- we heard the
04:44:23 book and bookmark example, which I think is a great

04:44:26 example. A pointer is just like a bookmark in a book.
04:44:29 It's just a way to get to some part of the book very
04:44:32 quickly.
04:44:34 Q. Does a bookmark change the book?
04:44:36 A. It does not. And a pointer does not change a packet.
04:44:41 Q. Are you aware of any routines in the three accused
04:44:46 products that operate on representations of packets?
04:44:53 A. No. All the routines I've seen operate on packets, and
04:44:58 specifically ethernet packets.
04:45:08 MR. BURESH: If we go to the next -- if we could
04:45:11 go to the next slide, please. Thank you.
04:45:15 Q. (By Mr. Buresh) What do we have depicted here,
04:45:18 Dr. Jeffay?
04:45:18 A. This is a diagram that we've now seen many, many times.
04:45:23 It's our graphical representation of an ethernet packet.
04:45:27 And, in this case, it's an ethernet packet containing some
04:45:32 generic application data.
04:45:34 Q. Is a packet an identifiable thing?
04:45:39 A. Yes.
04:45:39 Q. Does it have a start and an end?
04:45:43 A. Yes, it absolutely has a start and an end.
04:45:46 Q. Is a packet a real thing?
04:45:53 A. Yes. I mean, as I say, it is the unit of transmission
04:45:58 on a network. And to show you that it's real -- I mean,
04:46:02 this is -- this is how you get your web pages and your

04:46:05 emails and your videos. I mean, they come to you in
04:46:08 packets. They are real things.
04:46:11 Q. How do you identify the outermost header of a packet?
04:46:18 A. Well, this is easy. The way we've been drawing them
04:46:22 here, you just look at what's the -- the header on the
04:46:24 left.
04:46:25 Q. Does it depend?
04:46:28 A. I'm sorry?
04:46:29 Q. Does it depend?
04:46:30 A. Depend on what?
04:46:31 Q. I mean, can you just see the outermost header?
04:46:37 A. Yes, yes. No, it doesn't depend on -- it just depends
04:46:41 on what's the outermost header. I mean, it's -- it's
04:46:44 whatever the header on the left is.
04:46:48 MR. BURESH: I'd like to go, Mr. Palisoul, now to
04:46:50 the '683 patent, Column 1, and let's go to Lines 22 -- 24
04:46:58 to 44. Thank you.
04:46:59 Q. (By Mr. Buresh) Now, does the asserted patent describe
04:47:01 the process of converting packets?
04:47:03 A. It does.
04:47:09 Q. And is that depicted on the screen, that process?
04:47:12 A. Yes.
04:47:12 Q. And where would the jury find this description in the
04:47:15 patent -- in the patent?
04:47:17 A. Well, I think we don't have the -- the line numbers.

04:47:27 So this is Column 1, so it's the first page of the
04:47:29 specification. It's the first paragraph in the Background
04:47:32 section.
04:47:34 Q. Could you describe for us what this -- this section of
04:47:39 the patent in the Background section -- how does it
04:47:41 describe the converting process?
04:47:43 A. Okay. So the invention is described in the patent in
04:47:50 terms of end systems, so systems that are sending data and
04:47:54 systems that are -- two systems that are communicating
04:47:57 directly with one another.
04:47:58 And the highlighted text here is describing the
04:48:01 conversions that are required to send data from one
04:48:03 computer to the other.
04:48:04 And in this case, they're trying to send what they
04:48:10 call bitmap data. And as Dr. Almeroth said, you can think
04:48:13 of that as like an image. So you encrypt -- in this case,
04:48:17 they compress the data, they encrypt it, and then they
04:48:22 perform a series of conversions to transmit the data.
04:48:27 And the conversions consist of putting these
04:48:29 headers onto -- to build a packet. You place a TCP header
04:48:34 to get in a TCP format and then an IP header for an IP
04:48:38 format, and so on.
04:48:42 Q. What would this process look like?
04:48:44 MR. BURESH: If we go to the next slide, please.
04:48:46 A. Okay. Well, this is a slide that I sort of adapted

04:48:50 from my lecture notes. So we have two computers at the
04:48:53 bottom. The one on the right is going to send data to the
04:48:56 one on the left. And you start with an application that
04:48:59 has this bitmap data that it wants to transmit.
04:49:02 And as the application passes the data through the
04:49:07 operating system, the date -- we have the series of
04:49:12 conversions that are listed here by the downward arrows.
04:49:15 You convert into a TCP format by adding a TCP header, then
04:49:20 an IP format by adding an IP header, and then, finally, an
04:49:24 ethernet format by adding an ethernet header.
04:49:28 And then at this point, we have an ethernet
04:49:31 packet, and we can transmit it to the receiver.
04:49:34 Q. (By Mr. Buresh) And in each of these circumstances
04:49:35 that you had just had underlined there, is it the outermost
04:49:39 header that defines the format of the packet?
04:49:40 A. Yes.
04:49:44 Q. Is that why outermost header is important?
04:49:46 A. Yes.
04:49:47 Q. Now, what happens after this patent -- packet is sent
04:49:59 across the network?
04:49:59 A. Essentially, the process will reverse itself. The
04:50:03 figure will stay the same. I'm just going to flip the
04:50:08 arrows from going down to up.
04:50:09 Q. And is this described in the patent?
04:50:11 A. It is.

04:50:11 MR. BURESH: If we could go back to the '683
04:50:13 patent, Mr. -- you're there. Thank you.

04:50:16 Q. (By Mr. Buresh) What does the '683 patent say about
04:50:23 converting on the receiving side?

04:50:24 A. It says essentially what I just said, which is that the
04:50:30 conversions that happen at the sender will happen at the
04:50:33 receiver, just in the reverse order.

04:50:36 Q. On the sending side, you're adding one after the other;
04:50:42 is that correct?

04:50:42 A. That's correct.

04:50:42 Q. And what is the reverse of adding headers?

04:50:45 A. Removing headers.

04:50:51 MR. BURESH: Now if we could go back to your
04:50:53 demonstrative and look at the receiving side.

04:50:54 Q. (By Mr. Buresh) What's happening here?

04:50:56 A. So these are the conversions required to ultimately get
04:51:00 the bitmap data to the application. It comes in as a full
04:51:05 ethernet packet. It gets converted by deleting the
04:51:08 ethernet header to an IP packet. Delete the IP header, you
04:51:14 get your TCP packet. Delete the TCP header, you now just
04:51:19 have bitmap data for the application.

04:51:21 Q. And why are these computers, depicted on Slide 43, why
04:51:27 are they doing this conversion process?

04:51:29 A. They're doing this conversion process because, again,
04:51:33 as described in the specification of the patents, this is

04:51:38 technology that's targeted towards end systems where you
04:51:42 start with bitmap data generated by an application, and you
04:51:47 want to end up with bitmap data and a receiving
04:51:50 application. And the sender doesn't want to deal with
04:51:54 headers, and the receiving application doesn't want
04:51:57 headers.

04:51:57 So we need the headers to get it across the
04:52:00 network. That's a necessity. But we do these conversions
04:52:04 because that's what's required to transmit it. And we
04:52:08 deconvert at the receiver because the receiving application
04:52:13 doesn't want the headers.

04:52:14 Q. Now, do the NetScout products do this conversion
04:52:18 process?

04:52:18 A. No.

04:52:18 Q. Why not?

04:52:19 A. So the NetScout products, remember they're -- they're
04:52:22 not senders and receivers, they're in the middle of the
04:52:25 network. And, in particular, what they're doing is trying
04:52:31 to provide, I think they call it awareness of what's going
04:52:37 on in the network.

04:52:38 And one of the things that they will all do is
04:52:41 they all process full ethernet packets. And the
04:52:44 InfiniStream and the GeoProbe products, for example, will
04:52:46 store packets exactly as received because you need to do
04:52:49 that in order to do troubleshoot and debug problems.

04:52:55 So you do not want to be taking off headers
04:52:57 because the headers are going to have valuable information
04:52:59 for your statistics and for future troubleshooting.
04:53:04 Q. Now, did you analyze the NetScout products to determine
04:53:13 whether they operated on or converted packets that had an
04:53:16 outermost header TCP?
04:53:16 A. I did.
04:53:17 Q. What type of information did you consider?
04:53:19 A. The same information that I referenced previously; the
04:53:23 NetScout documents, the deposition testimony of persons
04:53:28 knowledgeable about the systems, and then, of course, the
04:53:31 source code.
04:53:32 Q. After reviewing that information, what did you
04:53:38 conclude?
04:53:39 A. I concluded that the accused products do not operate on
04:53:45 packets whose outermost header is a TCP header, and they do
04:53:48 not convert packets from a TCP format into another format.
04:53:54 Q. Starting with the InfiniStream --
04:53:56 MR. BURESH: If we can go to the next slide,
04:53:59 please.
04:53:59 Q. (By Mr. Buresh) What happens to a packet during
04:54:01 processing in the InfiniStream?
04:54:03 A. During the processing, the InfiniStream will create
04:54:12 this data structure that's called an F -- XMIB that's
04:54:18 illustrated here. And the XMIB will contain a series of

04:54:22 pointers to interesting locations in the packet.
04:54:26 So you get an ethernet packet in this case with
04:54:29 generic application data, and this is our book and bookmark
04:54:33 analogy. The -- the packet here -- the ethernet packet is
04:54:36 the book. And, in this case, we're just putting four
04:54:40 bookmarks into the book so we can find places in the book
04:54:43 quickly when we do the actual processing of the packet.
04:54:46 Q. What parts of this packet are stored in memory?
04:54:51 A. The whole packet, the whole ethernet packet is stored
04:54:54 in memory.
04:54:56 Q. And how do the processing routines have access to this
04:55:00 whole packet that's stored in memory?
04:55:01 A. Through this XMIB structure. With this XMIB structure,
04:55:06 you can -- you can quickly access any part of the packet
04:55:10 that you like.
04:55:10 Q. In what way, if any, do pointers change the packet?
04:55:14 A. Pointers do not change the packet at all. If you put a
04:55:19 bookmark in a book, you're not changing the book.
04:55:22 Q. Did you hear Mr. Barrett's testimony with respect to
04:55:27 the InfiniStream?
04:55:27 A. I did.
04:55:27 Q. And was your review of the source code consistent with
04:55:32 all of his testimony?
04:55:32 A. Yes.
04:55:33 Q. What happens to the -- the packet that we have depicted

04:55:42 here at the end of processing in the InfiniStream?
04:55:45 A. Ultimately, their goal is to generate statistics. And
04:55:50 then, as Mr. Barrett said, they will store the packet onto
04:55:54 disks so that if a problem -- if they become aware of a
04:55:59 problem, they can go back retrospectively, look at these
04:56:05 packets to try and debug that problem.
04:56:07 Q. Now, at any point in the process within the
04:56:12 processing -- at any point within the processing in the
04:56:16 InfiniStream, does the outermost header of a received
04:56:23 packet ever change?
04:56:23 A. No, it's always -- it's always stored as an ethernet
04:56:26 packet. It's always processed as an ethernet packet. And
04:56:28 the outermost header is always an ethernet header.
04:56:32 Q. In the InfiniStream, is there ever a packet that has an
04:56:37 outermost header of TCP?
04:56:40 A. No.
04:56:46 MR. BURESH: If we can go to the next slide,
04:56:49 please.
04:56:49 Q. (By Mr. Buresh) What's depicted here, Mr. --
04:56:53 Dr. Jeffay?
04:56:54 A. This is our same friendly ethernet packet now within
04:57:01 the context of the GeoProbe system. And here we're showing
04:57:04 the structure that the GeoProbe system uses to implement
04:57:07 its bookmarks. It does it using a slightly different
04:57:10 method.

04:57:11 Q. When a packet is received into the GeoProbe product,
04:57:15 what part of the packet is stored in memory?

04:57:20 A. The entire -- the entire ethernet packet is stored in
04:57:24 memory.

04:57:27 Q. Now, these offsets that come out of the packet
04:57:30 descriptor, where are they measured from?

04:57:33 A. They're all offsets from the start of the ethernet
04:57:37 packet.

04:57:38 Q. So how would a processing routine use the offsets to
04:57:44 locate information in the packet?

04:57:45 A. They use an operation that's called indexing, memory
04:57:51 indexing where you take a starting address, the start of
04:57:53 the ethernet packet, and you add the offset to it. And
04:57:57 then that gives you effectively the address of, for
04:58:00 example, the IP header or the TCP header or the start of
04:58:04 the application data.

04:58:08 Q. If the GeoProbe products were to change the header
04:58:13 structure of the packets it was receiving, what would that
04:58:16 do to the ability of the processing routines to find
04:58:19 information in these packets?

04:58:20 A. It would potentially break that ability because all of
04:58:27 these offsets are relative to the start of the ethernet
04:58:30 header. If the location of the header changes or the size
04:58:34 of the header changes, or if you take it off, all of those
04:58:37 offsets are going to break.

04:58:39 Q. When a system is using offsets like depicted here on
04:58:46 Slide 52 as in the GeoProbe products, does it change the
04:58:49 packet in any way?
04:58:49 A. No, the packet never changes. And, again, these are
04:58:53 offsets because it's just a different way of implementing a
04:58:56 bookmark. And, as I say, you can put 10 bookmarks in a
04:59:00 book, but it doesn't change the book.
04:59:06 Q. Now, did you hear Mr. Curtin's testimony from earlier
04:59:09 today with respect to the GeoProbe?
04:59:11 A. Yes.
04:59:11 Q. And was your review of the GeoProbe source code
04:59:14 consistent with Mr. Curtin's testimony?
04:59:15 A. Yes, it was.
04:59:17 Q. What happens to a packet in the GeoProbe after the
04:59:23 processing is complete?
04:59:24 A. The GeoProbe, again, is a product that's trying to give
04:59:29 network administrators some visibility in other networks,
04:59:33 so it will be stored in the unlikely event that there's --
04:59:37 they discover a problem later. They can effectively go
04:59:40 back in time, look at the packets that were received, and
04:59:43 try and understand the nature and the cause of that
04:59:45 problem.
04:59:49 Q. At any point during processing in the GeoProbe
04:59:52 products, Dr. Jeffay, does the outermost header of a packet
04:59:55 ever change?

04:59:56 A. No, it stays ethernet. And it has to stay ethernet
05:00:00 because, as I say, in both in InfiniStream and GeoProbe,
05:00:04 when they're saving the packets, they need to save all the
05:00:07 information in the packets because all of that information
05:00:10 is going to potentially be needed to debug a problem.
05:00:19 Q. In the GeoProbe products, is the outermost header of
05:00:21 any packet ever TCP?
05:00:23 A. No, it's always ethernet.
05:00:28 MR. BURESH: If we can go to the next slide,
05:00:30 please.
05:00:30 Q. (By Mr. Buresh) Now, this is the abuf and the pbuf
05:00:35 from the Arbor product; is that correct?
05:00:36 A. Correct. This slide unfortunately lost its title, but
05:00:39 this -- this is -- this is for the Arbor -- Arbor systems.
05:00:42 Q. Now, when a packet comes into the Arbor systems, is it
05:00:47 stored to memory?
05:00:47 A. It is.
05:00:48 Q. How much of it?
05:00:51 A. The entire packet, start to finish, the entire ethernet
05:00:55 packet is stored in memory.
05:00:56 Q. And what happens to the packet at that point?
05:00:58 A. Well, again, these products are sort of similar at some
05:01:04 level. They build a structure. We've heard about this
05:01:06 one. This is -- it's actually two -- two structures,
05:01:09 what's called an abuf and a pbuf. And, again, these

05:01:13 structures just simply are pointers to interesting parts of the ethernet packet that's in memory.

05:01:17 Q. Do pointers change the packet in any way?

05:01:22 A. No, the bookmarks in the book do not change the book.

05:01:26 Q. What is the impact of a pointer on the outermost header of a packet?

05:01:30 A. There is no impact on it. The pointer does not impact the outermost header.

05:01:38 Q. Did you review the Arbor source code?

05:01:46 A. I did.

05:01:46 Q. And did you listen to Dr. Dawson's testimony from earlier today?

05:01:50 A. I did.

05:01:50 Q. Are those two consistent, the source code and his testimony?

05:01:53 A. Yes, I looked at the exact same source code that he's -- that he was talking about.

05:01:57 Q. In the Arbor products, what happens to a packet after it's done being processed?

05:02:03 A. So let's remember, the Arbor products, they're different. They're the security products, so they're looking at traffic as it's flowing from sender to the receiver, just trying to understand whether or not they should drop it because it's potentially malicious or let it go through because it's -- it's good traffic.

05:02:23 So after the processing is done, assuming it's a
05:02:26 good ethernet packet, it just simply is sent on its way
05:02:30 toward its intended destination.
05:02:36 Q. In what way -- let me ask you this: Was the fact that
05:02:43 you saw pointers and offsets in the source code in the
05:02:46 NetScout products, was that relevant to your analysis?
05:02:48 A. Well, given that I had understood that these pointers
05:02:55 were the basis for an infringement theory, yes. It was --
05:02:58 it was relevant for my analysis.
05:03:00 Q. What is the fact that the NetScout products used
05:03:03 pointers and offsets, what does that tell you about the
05:03:07 converting requirements with respect to the claims in this
05:03:10 case?
05:03:11 A. It precisely tells me there is no converting. If there
05:03:18 were converting, there would be something different in
05:03:21 memory other than the original packet as it was received.
05:03:25 Q. Now, did you hear Dr. Almeroth's testimony from
05:03:28 yesterday?
05:03:29 A. Yes.
05:03:31 Q. And he concluded the opposite, that the NetScout
05:03:36 products do convert. Did you hear that?
05:03:38 A. I did.
05:03:39 Q. What did you think about his analysis?
05:03:45 A. Well, obviously, I disagree. It's -- it strikes me as
05:03:52 a little not non-standard. He spoke a lot about

05:03:56 representations of packets.

05:04:01 And at the end of the day, all that's in the
05:04:03 accused systems are just the packets. Yes, they have
05:04:06 pointers to them. But they just have the ethernet packets
05:04:10 as they were received off the wire.

05:04:13 Q. Before we leave the Arbor products, in the Arbor
05:04:27 products is there ever a packet whose outermost header is
05:04:35 TCP?

05:04:35 A. No.

05:04:36 Q. Okay. Before moving forward, I need to go back a step
05:04:39 to the sequence of routines analysis. Do you recall
05:04:43 discussing that a little while ago?

05:04:44 A. Yes.

05:04:45 MR. BURESH: Can you pull up the Court's claim
05:04:50 construction of sequence of routines, please?

05:05:27 Q. (By Mr. Buresh) Now, earlier, we were discussing the
05:05:30 sequence of routines; do you recall that?

05:05:31 A. I do.

05:05:31 Q. And you testified a number of times that in the
05:05:36 NetScout products, the sequence of routines were created
05:05:38 before a first packet of a message?

05:05:41 A. That's -- that's correct.

05:05:43 Q. Okay. And when you were making that statement, were
05:05:46 you applying the Court's claim construction?

05:05:48 A. Yes, I was.

05:05:49 Q. Is it equally true that an ordered arrangement of
05:05:54 routines is not created in the NetScout products prior to
05:05:57 receiving a first packet of the message?
05:05:59 A. I'm sorry, the cough through me off. Could you maybe
05:06:03 just repeat your question?
05:06:05 Q. Yeah. I said, is it equally true that an ordered
05:06:07 arrangement of software routines is not created before
05:06:10 receiving a first packet of the message in the NetScout
05:06:16 products?
05:06:16 A. They -- they are created --
05:06:19 Q. I said that backwards. My bad.
05:06:22 THE COURT: Why don't you withdraw that question
05:06:23 and re-ask it.
05:06:24 MR. BURESH: I'm going to go ahead and withdraw
05:06:27 that one, Your Honor. Thank you.
05:06:28 Q. (By Mr. Buresh) Dr. Jeffay, is it equally true that
05:06:32 the ordered arrangement of routines in the NetScout
05:06:34 products is created before receiving a first packet of a
05:06:37 message?
05:06:37 A. That I can agree with.
05:06:38 Q. Thank you.
05:06:43 Is it equally true that the processing paths in
05:06:47 the NetScout products are created before receiving a first
05:06:50 packet of a message?
05:06:50 A. Yes. They're -- they're statically created when the

05:06:55 code was designed and written.
05:06:56 Q. And that is true with respect to all three NetScout
05:06:59 products; the Arbor, the InfiniStream, and the GeoProbe?
05:07:02 A. That is correct.
05:07:07 MR. BURESH: Mr. Palisoul, can you pull up the
05:07:09 next demonstrative, please? Thank you. This is fine.
05:07:26 Q. (By Mr. Buresh) And, Dr. Jeffay, we've looked at the
05:07:29 executed Transmission Control Protocol requirement and the
05:07:33 convert one or more packets having a TCP format into a
05:07:36 different format requirement. Are those concepts present
05:07:39 in each asserted claim?
05:07:41 A. Yes. They are illustrated here for the -- what are
05:07:47 called the independent claims, the claims that do not
05:07:49 reference another claim. And the highlighting shows that
05:07:53 one of those two claimed phrases is in each of the
05:07:57 independent claims.
05:08:01 Q. Have you heard some testimony about reassembly in this
05:08:07 case?
05:08:08 A. I did.
05:08:10 Q. Does reassembly in the NetScout products satisfy the
05:08:17 execute a Transmission Control Protocol or convert one or
05:08:21 more packets having a TCP format into a different format?
05:08:26 Does reassembly satisfy either of those limitations?
05:08:29 A. No.
05:08:31 Q. I want to look at the Arbor product first. Do the

05:08:42 Arbor products do any reassembly?
05:08:44 A. No, they do not.
05:08:45 Q. How do the Arbor products process packets that arrive
05:08:51 out of order?
05:08:52 A. So packets can arrive out of order because they can
05:08:58 take different routes between the sender and the time they
05:09:01 hit the Arbor box. And the Arbor box will simply hold on
05:09:04 to packets until they have arrived -- until it has enough
05:09:09 that they can be transmitted in order.
05:09:15 Q. And how do they accomplish that?
05:09:17 A. They have a data structure, a list -- they do it
05:09:21 exactly like you and I would do it. If I receive Packet
05:09:25 No. 3 and I was expecting Packet No. 1, I would just make a
05:09:30 list. Say, I have 3, looking for 1.
05:09:32 Then if they receive 2, they -- No. 2 on the list,
05:09:35 still waiting for No. 1. And then, finally, when No. 1
05:09:39 comes in, you'd recognize, okay, now I have them in order.
05:09:42 Now I can send them to the destination. And then you send
05:09:46 them in order.
05:09:47 Q. And I'm just signalling to slow down a little bit.
05:09:53 A. Yes. I apologize to everyone.
05:09:55 Q. In the Arbor products, does that re-ordering list that
05:09:58 you've just described, does that change the headers on the
05:10:01 received packets in any way?
05:10:03 A. No. And it -- and it can't change the headers because,

05:10:08 remember, the packets are ultimately going to be sent to
05:10:12 the intended destination. And if you change the headers,
05:10:15 they're potentially not going to get to the destination.
05:10:18 Q. How do the GeoProbe products process packets that
05:10:22 arrive out of order?
05:10:23 A. It uses a similar technique, essentially buffering the
05:10:31 packets, holding on to the ones that are out of order,
05:10:34 waiting until the in order ones arrive.
05:10:38 Q. Does that process result in any change in the packets
05:10:43 in the GeoProbe system?
05:10:43 A. No. You're just -- you're just storing them
05:10:47 temporarily. And, again, ultimately they're going to be
05:10:49 all written to disk. As we've talked about, you want to
05:10:51 write them -- every piece of -- every byte of data in a
05:10:54 header, you want to write to disk so that you can
05:10:58 troubleshoot later problems. So -- so you don't change --
05:11:04 you can't change the headers.
05:11:05 Q. Did you hear Mr. Curtin explain reassembly in the
05:11:10 GeoProbe products?
05:11:11 A. I did.
05:11:11 Q. And can you briefly describe that process?
05:11:15 A. Sure. So reassembly results in -- in making this --
05:11:23 what he calls synthetic ethernet packet. So you take
05:11:27 headers from the first packet, unmodified, and then you
05:11:32 concatenate onto them the payloads from the packets that

05:11:36 you're trying to reassemble. And you concatenate the
05:11:39 payloads in the right order.
05:11:41 Q. Are any headers changed on any packets in that process?
05:11:45 A. No. The headers on the synthetic packet are exactly
05:11:49 the headers on the first packet.
05:11:53 Q. What about the InfiniStream, does it do reassembly?
05:11:56 A. Yes, it does a form of -- of reassembly.
05:12:00 Q. And there was some discussion earlier in the case about
05:12:03 an errata sheet. Do you remember that?
05:12:05 A. Yes. Deposition errata, yes.
05:12:12 Q. Yes.
05:12:13 A. Yes.
05:12:13 Q. Did you look into that -- the source code for the
05:12:15 InfiniStream? Does it show you one way or another whether
05:12:18 there is reassembly on HTTP traffic?
05:12:20 A. Yes. I found no code for reassembly of HTTP traffic.
05:12:26 So I believe that the product does not do HTTP reassembly.
05:12:32 Q. For the protocols that it does provide reassembly for,
05:12:36 can you briefly describe how the InfiniStream does
05:12:38 reassembly?
05:12:40 A. Sure. So it's -- it's a little different because there
05:12:44 are no headers involved. You're simply taking the -- the
05:12:50 individual payloads and concatenating them together in the
05:12:53 correct order so that you effectively have, you know,
05:12:56 the -- the application data object that the application

05:13:00 transmitted. You have the full -- the full letter that was
05:13:02 in the envelope.

05:13:05 Q. Now, the packets that are involved in that process, are
05:13:09 the headers changed at all?

05:13:10 A. No, the original headers -- sorry, the original
05:13:14 ethernet packets that contain the data that was being
05:13:18 reassembled, those original packets never changed. They
05:13:22 stayed as ethernet packets, and they're ultimately flushed
05:13:25 to disk as ethernet packets.

05:13:32 Q. At any point during the reassembly process in the
05:13:36 InfiniStream, is there a packet that has TCP as its
05:13:38 outermost header?

05:13:39 A. No. InfiniStream, you only ever have -- for structures
05:13:46 that have headers, you only have ethernet as the outermost
05:13:49 header.

05:13:49 Q. Does the reassembly process in the InfiniStream result
05:13:52 in any conversion of the packets?

05:13:54 A. No, not in the required sense of the claims where
05:14:00 you're converting from a TCP format to another format.
05:14:05 There's -- there's no such conversion.

05:14:21 MR. BURESH: If we could pull up the claims again
05:14:21 with the execute convert limitations. Thank you.

05:14:25 Q. (By Mr. Buresh) Now, Dr. Jeffay, did the NetScout
05:14:28 products -- products -- I'm going to speak in a group now
05:14:31 to include the InfiniStream, the GeoProbe, and the Arbor

05:14:33 products -- do they operate on any packet whose outermost
05:14:38 header is TCP?
05:14:39 A. They do not.
05:14:44 MR. BURESH: Is it possible to overlay the Court's
05:14:46 claim construction on this one? I just need from "convert
05:15:19 limitations" down to the "execute limitations." Thank you.
05:15:34 Q. (By Mr. Buresh) And, Dr. Jeffay, starting over, you
05:15:36 see the execute a Transmission Control Protocol limitation?
05:15:40 A. Yeah. In the claims, yes.
05:15:41 Q. In the chart?
05:15:42 A. Oh, in the chart, yes, I see it in the chart, as well.
05:15:45 Q. And the Court's construction off to the right?
05:15:48 A. Yes.
05:15:48 Q. And do the NetScout products operate on any packets
05:15:52 whose outermost header is a TCP header?
05:15:54 A. No. The outermost header in the accused products is
05:15:58 always an ethernet header.
05:16:00 Q. In this highlighted claim construction here, do the
05:16:04 NetScout products ever convert packets from an outermost
05:16:08 header structure of TCP to another type of header
05:16:12 structure?
05:16:12 A. No. And the easiest way to see this is there never is
05:16:18 a packet with an outermost header structure that's TCP.
05:16:21 Q. And these limitations are in each of the asserted
05:16:27 claims?

05:16:27 A. Correct.
05:16:27 Q. In light of all your analysis of the NetScout source
05:16:33 code and the other documents and deposition testimony
05:16:36 you've seen and then heard, have you reached a conclusion
05:16:41 as to whether the NetScout products infringe any asserted
05:16:45 claim in this case?
05:16:45 A. Yes. My opinion is they do not infringe any asserted
05:16:50 claim, for at least the reason that they do not contain
05:16:54 either the converting or the execute -- either the
05:16:57 converting one or more packets limitations or the execute a
05:17:03 Transmission Control Protocol limitation.
05:17:06 Q. Okay.
05:17:07 MR. BURESH: If we could go to the next
05:17:09 demonstrative, please.
05:17:14 Q. And, Dr. Jeffay, looking at Claim 1 of the '683 patent,
05:17:19 can you explain which limitations are not satisfied by the
05:17:22 NetScout products?
05:17:23 A. Yes. If we look at Claim 1, what we're going to do is
05:17:31 look for the -- the elements, the -- the bits between the
05:17:36 green lines that either have the language sequence of
05:17:41 routines, execute a Transmission Control Protocol or
05:17:46 convert one or more packets. So we can start with the
05:17:50 convert -- sorry, the create limitation.
05:17:55 MR. BURESH: Advance the slide, please. Thank
05:17:56 you.

05:17:56 A. So this -- this claim element has the sequence of
05:18:01 routines limitation in it. And, in my opinion, the accused
05:18:06 products do not satisfy this limitation because they do not
05:18:09 meet the Court's definition for sequence of routines.
05:18:15 Q. (By Mr. Buresh) And which limitations of Claim 1 does
05:18:17 this impact?
05:18:18 A. This impacts the -- the limitation we're talking about,
05:18:24 the create limitation, as well as the next one, the store
05:18:30 the created path.
05:18:31 The -- the path is going to include the data
05:18:33 structure that indicates the sequence of routines. If
05:18:37 there's no sequence of routines, there's no data structure,
05:18:40 there's no path. So you can't store the created path.
05:18:42 Q. What about the next limitation to the bottom there?
05:18:48 A. The next limitation also references sequence of
05:18:52 routines, so that's one reason why we won't have that
05:18:55 limitation. But, in addition, it also references the
05:18:58 execute a transmission control. And it also references the
05:19:04 convert one or more packets element.
05:19:06 So according to the definitions the Court has
05:19:10 provided, in my opinion, the accused products do not
05:19:13 infringe -- do not contain this element because they
05:19:16 don't -- they don't satisfy the definitions that were just
05:19:19 on the screen.
05:19:19 Q. So, in your opinion, Dr. Jeffay, does NetScout infringe

05:19:28 Claim 1 of the '683 patent?

05:19:29 A. In my opinion, they do not.

05:19:32 MR. BURESH: Let's go to the next claim, please.

05:19:35 Q. (By Mr. Buresh) This is Claim 10 of the '683 patent.

05:19:37 And can you explain to the jury which limitations 05:19:39 are not satisfied by the NetScout products?

05:19:43 A. So, again, we're going to look through the text here to 05:19:51 find things related to the sequence of routines. And 05:19:52 remember, there were two forms of the sequence of routines 05:19:55 limitation. There was sequence of routines and also a list 05:19:59 of conversion routines.

05:20:01 So Claim 10 uses a different language, list of 05:20:04 conversion routines, but it has a very similar 05:20:07 construction. And the construction is not met, for the 05:20:11 reasons that we didn't have sequence of routines 05:20:19 limitation.

05:20:19 So this will knock out these four limitations 05:20:21 because the accused products do not have the list of 05:20:24 conversion routines.

05:20:25 Q. What about the bottom limitation?

05:20:29 MR. BURESH: If we can advance the slide.

05:20:32 Q. (By Mr. Buresh) Oh, the next limitation?

05:20:33 A. These ones reference the conversion element that we 05:20:37 talked about for Claim 1. And so this limitation is also 05:20:40 not going to be present because the accused products do not

05:20:43 satisfy the Court's definition of conversion because they
05:20:47 don't convert the outermost header structure from a
05:20:53 transport layer protocol header to another type of header
05:20:55 structure.

05:20:55 Q. So, in your opinion, Dr. Jeffay, does NetScout
05:21:02 infringe -- utilize the technology in Claim 10 of the '683
05:21:07 patent?

05:21:07 A. No. And maybe by way of explanation I could just
05:21:10 simply say -- to remind us all that to infringe, you have
05:21:13 to have -- as I understand it, you have to have each and
05:21:16 every element. So if you're missing one element, you do
05:21:19 not infringe. And, in this case, I'm saying that they're
05:21:21 missing five elements.

05:21:26 MR. BURESH: Let's go now to the '790 patent,
05:21:29 please.

05:21:29 Q. (By Mr. Buresh) This is Claim 1 from the '790 patent.
05:21:33 Which limitations of this claim are not present in
05:21:40 NetScout's products?

05:21:41 A. We're going to have -- the identifying process
05:21:44 limitations are not going to be present. The identify a
05:21:48 path limitation is not going to be present because it
05:21:50 references a sequence of two or more routines. And as I've
05:21:56 said, the accused products do not satisfy the Court's
05:21:58 construction of sequence of routines.

05:22:01 Q. What about the next limitation, Dr. Jeffay?

05:22:03 A. This one has both sequence of routines, the execute a
05:22:08 Transmission Control Protocol phrase, and the convert one
05:22:10 or more packets. And for the reasons that I've described
05:22:14 throughout my testimony, the accused products do not have
05:22:17 the sequence of routines, as defined by the Court; they do
05:22:20 not execute a Transmission Control Protocol, as defined by
05:22:24 the Court; and they do not convert the one or more packets,
05:22:32 as defined by the Court.

05:22:33 Q. Dr. Jeffay, what is your opinion with respect to
05:22:36 whether the NetScout products infringe Claim 1 of the '790
05:22:40 patent?

05:22:40 A. It's my opinion that they don't infringe because they
05:22:43 do not contain each and every element of Claim 1 of the
05:22:50 '790 patent.

05:22:50 MR. BURESH: If we could go to the '104 patent,
05:22:53 please.

05:22:53 Q. (By Mr. Buresh) Now, looking at Claim 1, first,
05:22:59 Dr. Jeffay, which limitations in Claim 1 are not present in
05:23:04 NetScout's products?

05:23:04 A. These would be all of the limitations that reference
05:23:07 sequence of two or more routines and executing the
05:23:11 Transmission Control Protocol. So the first one in Claim 1
05:23:14 would be this identifying using the key value limitation.

05:23:25 Q. What other limitations are not present from Claim 1?

05:23:28 A. The next create a path limitation is not present

05:23:33 because it, again, references the sequence of two or more
05:23:36 routines. And, as I've said, the accused products do not
05:23:39 contain the sequence of two or more routines, as that
05:23:43 phrase has been defined by the Court.

05:23:50 Q. Okay. Now, with respect to Claim 3 -- stop there.

05:23:55 With respect to Claim 1, before we move on, what
05:23:59 is your opinion with respect to whether NetScout's products
05:24:03 infringe Claim 1 of the '104 patent?

05:24:05 A. It's my opinion that the accused NetScout products do
05:24:09 not infringe Claim 1 because to infringe, you must have
05:24:13 each and every element of the claims present. And as I've
05:24:19 argued, the accused products do not contain at least the
05:24:23 three limitations that I've X'd out.

05:24:30 Q. With respect to Claim 3, do the NetScout products
05:24:30 infringe Claim 3 of the '104 patent?

05:24:33 A. They do not.

05:24:35 Q. Why not?

05:24:35 A. So this gets a little more detailed in the way that
05:24:41 patents work. But these are dependent claims. And you can
05:24:44 see that by the fact that they -- each one of them
05:24:47 references a prior claim. So Claim 3 references Claim 1,
05:24:52 and Claim 4 references Claim 3.

05:24:55 And, as I understand it, what this means is that
05:24:58 to understand infringement of Claim 3, you have to bring in
05:25:02 all of the elements of Claim 1. So if elements of Claim 1

05:25:07 are missing, you cannot infringe Claim 3.
05:25:10 And it's a similar story for Claim 4. Claim 4
05:25:18 depends, as we say, from Claim 3 which depends from
05:25:18 Claim 1. So to infringe Claim 4, you would have to have
05:25:22 all of the elements shown on this slide. And because we're
05:25:27 missing, for example, at least the three elements from
05:25:30 Claim 1, you can't infringe Claim -- the accused products
05:25:33 cannot infringe Claim 4.
05:25:35 Q. Now, Dr. Jeffay, do any of the three NetScout products,
05:25:39 the InfiniStream, the Arbor, or the GeoProbe, infringe any
05:25:44 asserted claim of Mr. Balassanian's patents that he's
05:25:47 asserted here in this case?
05:25:48 A. In my expert opinion, they do not.
05:25:53 MR. BURESH: Nothing further, Your Honor.
05:25:54 THE COURT: You pass the witness?
05:25:55 MR. BURESH: I do.
05:25:56 THE COURT: All right. Counsel, approach the
05:26:01 bench, please.
05:26:02 (Bench conference.)
05:26:10 THE COURT: Do you have an estimate on your time
05:26:18 for cross, Mr. Hosie?
05:26:19 MR. HOSIE: I think it will be about an hour and
05:26:22 15 minutes.
05:26:23 THE COURT: Well, we'll get as far as we can. We
05:26:26 won't finish it today, but we're going to have to get into

05:26:28 it.

05:26:29 MR. HOSIE: I appreciate that, Your Honor.

05:26:30 THE COURT: Let's proceed.

05:26:32 (Bench conference concluded.)

05:26:35 THE COURT: All right. Let's proceed with
05:26:36 Plaintiff's cross-examination of the witness.

05:26:50 MR. HOSIE: Thank you, Your Honor.

05:26:50 CROSS-EXAMINATION

05:26:50 BY MR. HOSIE:

05:26:50 Q. Dr. Jeffay, you testified, I think, four times that
05:26:58 Implicit's patents were restricted to end systems, e-n-d.
05:27:03 Do you recall saying that?

05:27:04 A. I -- I don't believe I ever said the word restricted.
05:27:08 I believe what I said is that's what's described in the
05:27:13 specification.

05:27:13 Q. All right, sir. For the jury, what is an end system?

05:27:15 A. Good, good question. End systems refer to the
05:27:18 computers that generate the data and the computers that
05:27:21 ultimately consume the data. So client server, your web
05:27:27 browser, that's one end system; a web server, that's
05:27:31 another end system.

05:27:32 Q. Is it your testimony here today, sir, that the Implicit
05:27:35 claims would not apply to a server sitting between two
05:27:39 other computers?

05:27:40 A. I haven't offered that testimony.

05:27:43 Q. You do not believe that yourself, do you?
05:27:45 A. No, I don't believe that.
05:27:47 Q. All right, sir. So it really doesn't matter from the
05:27:50 purposes of the claims and infringement whether -- I'm
05:28:00 sorry, strike that.
05:28:04 MR. HOSIE: May we have the Claim 1 of the '683
05:28:07 up, please.
05:28:09 Q. (By Mr. Hosie) Sir, this is Claim 1 of the '683. Do
05:28:15 you see anywhere in this claim the words end systems, e-n-d
05:28:19 systems?
05:28:20 A. No.
05:28:20 Q. All right.
05:28:21 MR. HOSIE: Thank you.
05:28:22 Q. (By Mr. Hosie) Now, you were in court for
05:28:30 Dr. Almeroth's testimony, I believe you said?
05:28:31 A. Yes, I was.
05:28:31 Q. And you know Professor Almeroth?
05:28:34 A. I do.
05:28:34 Q. You personally know him?
05:28:36 A. I do.
05:28:36 Q. And you respect him as a computer scientist?
05:28:38 A. Yes.
05:28:39 Q. And you respect him as a professor?
05:28:41 A. I -- I respect him as a computer scientist. We don't
05:28:46 really interact as faculty peers.

05:28:49 Q. Fair enough. You believe he does good and honorable
05:28:52 computer science work?
05:28:53 A. He has done some fine research in the past.
05:28:55 Q. You believe he's a person of integrity?
05:28:58 A. As best I can tell.
05:28:59 Q. All right, sir. And you've read some of his research
05:29:02 papers, I'm sure?
05:29:03 A. I have.
05:29:03 Q. And you found them good, I'm sure?
05:29:05 A. Yeah.
05:29:06 Q. All right, sir. And you heard Dr. Almeroth describe
05:29:13 various things he was asked to do in this case?
05:29:15 A. I did.
05:29:16 Q. And one of the things, of course, was to analyze
05:29:18 infringement?
05:29:18 A. Yes, absolutely.
05:29:19 Q. Which you, too, have done?
05:29:21 A. Correct.
05:29:21 Q. Fair to say that you land on different sides of that
05:29:25 line?
05:29:25 A. Well, I think that's quite fair.
05:29:26 Q. I think that's quite fair. But Dr. Almeroth also was
05:29:32 asked to assess acceptable non-infringing alternatives,
05:29:38 correct?
05:29:38 A. I don't recall if that was part of his task. I know he

05:29:41 certainly testified about that.

05:29:42 Q. And is that something you have assessed, sir,
05:29:46 non-infringing alternatives?

05:29:46 A. I have not.

05:29:47 Q. So you have no basis to agree or disagree with
05:29:51 Dr. Almeroth's analysis on that score?

05:29:53 A. Well, I believe Dr. Analysis's analysis --
05:29:59 Dr. Almeroth -- excuse me, Dr. Almeroth's analysis was
05:30:01 simply that NetScout didn't offer any non-infringing
05:30:06 alternatives.

05:30:06 Q. Have -- and do you have any reason to disagree with
05:30:08 that?

05:30:09 A. I -- I do not.

05:30:12 Q. And you also heard Dr. Almeroth analyze licenses for
05:30:23 technical comparability -- comparability?

05:30:24 A. Yes, yes, I did hear that.

05:30:26 Q. And is that something you looked at, sir?

05:30:28 A. I did not.

05:30:29 Q. So --

05:30:30 A. I was not asked to do that, sorry.

05:30:32 Q. You have no basis to disagree with his conclusions on
05:30:36 that score, do you?

05:30:36 A. Correct. I'm not addressing his conclusions.

05:30:39 Q. All right.

05:30:52 MR. HOSIE: If we could have Figure 15 of the '683

05:30:58 patent up, sir.

05:30:59 Q. (By Mr. Hosie) Do you recognize this, sir?

05:31:05 A. I do.

05:31:06 Q. This is a flow chart, is it not, sir?

05:31:08 A. It is.

05:31:10 Q. This is Figure 15 from the '683 patent, correct?

05:31:14 A. Yes, that's correct.

05:31:16 Q. You said, as I recall, it takes time to go through the
05:31:21 steps of a flowchart. Do you recall saying that?

05:31:23 A. I did.

05:31:24 Q. That's your testimony and opinion, correct?

05:31:27 A. Yes, absolutely true.

05:31:29 MR. HOSIE: All right. Your Honor, may we seal
05:31:30 the courtroom. I'm about to move into a confidential
05:31:33 document.

05:31:34 THE COURT: All right. Based on counsel's
05:31:37 request, I'll order the courtroom sealed at this time.

05:31:40 Anyone present who is not subject to the existing
05:31:43 protective order that's been entered in this case should
05:31:46 excuse themselves and remain outside the courtroom until
05:31:49 it's reopened and unsealed.

05:31:50 We're sealed, for the record, counsel.

05:31:50 (Courtroom sealed.)

05:31:50 (This portion of the transcript is sealed and
05:31:50 filed under separate cover as Sealed Portion

05:32:41 No. 8.)

05:32:41 (Courtroom unsealed.)

05:32:42 THE COURT: We're back on the record and unsealed.
05:32:46 Let's proceed.

05:33:06 MR. HOSIE: If we may have '683 claim up, Claim 1.

05:33:06 Q. (By Mr. Hosie) Line 25 starts with the word create.
05:33:09 Do you see that, sir?

05:33:09 A. I do.

05:33:10 Q. And then it goes on to say, a sequence of routines,
05:33:13 correct?

05:33:13 A. Yes.

05:33:21 Q. And you're creating a path here, correct?

05:33:23 A. Yes.

05:33:24 Q. Because it says create, da, da, da, da, a path?

05:33:30 MR. HOSIE: Can we have "a path" highlighted?

05:33:32 A. I'm sorry, what's the question?

05:33:34 Q. (By Mr. Hosie) It says create a path?

05:33:35 A. Yes, absolutely.

05:33:37 Q. And then you store that created path?

05:33:39 A. Correct.

05:33:39 Q. And creating a path, in your mind, does that include a
05:33:44 system that just has programming logic linking modules but
05:33:50 no ability to actually process anything as a path?

05:33:53 A. I'm sorry, but I don't understand that -- that
05:33:57 question.

05:33:57 Q. Let me put it again. Assume a system that has program
05:34:01 code that links modules together but has no ability to
05:34:07 actually process traffic, does that system have a path
05:34:12 within the meaning of this claim as you read it, sir?
05:34:15 A. I can't -- I can't envision the system that you're -- a
05:34:20 system that links things together but cannot process
05:34:24 anything?
05:34:26 Q. Correct.
05:34:27 A. And then you're asking me what -- what exactly?
05:34:29 Q. Is that -- is that linked system that can process
05:34:32 nothing in the real world as it created a path, sir, within
05:34:36 the meaning of this claim?
05:34:37 A. I don't know that there's enough information in your
05:34:44 question for me to really tell.
05:34:44 Q. So you can't answer it as I've phrased it?
05:34:47 A. No, I'm sorry, I don't think I can.
05:34:52 Q. Now, in the NetScout systems, a path is built
05:34:59 incrementally as the process runs, correct?
05:35:05 A. A path is built incrementally as the -- no, I would --
05:35:10 I would disagree.
05:35:11 Q. Well, let's go back to the pair of green --
05:35:18 Exhibit 3 -- oh, I'm going to have to seal the courtroom
05:35:21 again.
05:35:22 Let me -- let me walk you through it at a higher
05:35:25 level.

05:35:25 So a packet comes into NetScout's system, correct?
05:35:28 A. Sure.
05:35:28 Q. The system looks at the packet?
05:35:30 A. Okay.
05:35:31 Q. And it has to look at the packet to see what's inside
05:35:34 the packet?
05:35:38 A. Sure. As -- as part of the processing of the packet,
05:35:40 it will certainly look into the packet.
05:35:42 Q. And then it processes the various layers of the
05:35:46 processing stack?
05:35:49 A. For, yes, some of the products, yes, that will happen.
05:35:54 Q. And it goes through various branching decision trees,
05:36:01 if/then/else statements?
05:36:04 A. Sure. Those are the analogs of the junctures and the
05:36:08 plumbing diagrams that --
05:36:08 Q. Right. So --
05:36:09 A. -- we had.
05:36:09 Q. Thank you. I didn't mean to interrupt, sir.
05:36:11 So the system first looks at the packet, and it
05:36:14 says, okay, I've got to make a decision, what do I do?
05:36:18 A. Okay.
05:36:20 Q. And then it makes that decision and it goes on and
05:36:23 says, okay, I've got to make another decision of what I
05:36:27 would do?
05:36:27 A. Well, I wouldn't -- I wouldn't characterize it that

05:36:30 way.
05:36:30 Q. Well, is that inaccurate, sir?
05:36:32 A. I think it is.
05:36:32 Q. Well, doesn't the system do that kind of what do I do?
05:36:39 What do I do? What I do I do step-by-step-by-step?
05:36:44 A. As I say, I wouldn't characterize it that way, and I'm
05:36:49 happy to explain if you like.
05:36:51 MR. HOSIE: Perhaps I better go back to PTX-381,
05:36:51 Your Honor. I'll need to close the courtroom again if the
05:36:58 Court --
05:36:58 THE COURT: Based on counsel's request, I'll order
05:36:58 the courtroom sealed at this time. Those present not
05:37:00 subject to the protective order should excuse themselves
05:37:03 until the courtroom is unsealed.
05:37:08 (Courtroom sealed.)
05:37:08 (This portion of the transcript is sealed and
05:37:08 filed under separate cover as Sealed Portion
05:37:08 No. 9.)
05:37:08 (Courtroom unsealed.)
05:48:16 MR. HOSIE: Thank you, Your Honor.
05:48:17 THE COURT: Let's continue.
05:48:19 Q. (By Mr. Davis) Earlier in your testimony, about six or
05:48:20 seven minutes ago, you said, and I think I can quote:
05:48:21 These systems don't think for themselves. Do you recall
05:48:23 saying that, or words to that effect?

05:48:25 A. I think I was making a comment that -- that you were
05:48:30 using anthropomorphic language, I think. But, roughly,
05:48:35 that's fair.
05:48:35 Q. Okay. And what did you mean by these computers don't
05:48:40 think for themselves?
05:48:41 A. Well, I don't recall the context of the individual
05:48:44 question, but I believe you were describing the processing
05:48:48 in the Arbor system in terms of what people do as opposed
05:48:51 to what computers do.
05:48:53 Q. All right, sir. You agree with me that a computer is
05:48:55 just a machine?
05:48:56 A. It is indeed a machine.
05:48:57 Q. Have you heard the phrase deterministic conduct in
05:49:02 connection with computers?
05:49:03 A. Deterministic, yes. Conduct, no.
05:49:06 Q. Fair point. What does the phrase deterministic mean to
05:49:10 you when used in the context of a computing system?
05:49:14 A. Deterministic means it's known what's going to happen,
05:49:18 and it's in contrast to probabilistic, which means there's
05:49:22 some chance.
05:49:23 Q. And aren't all computer systems, sir, deterministic?
05:49:28 A. They're supposed to be -- the computer systems that
05:49:32 we're talking about here, yes, they're -- they
05:49:34 deterministically execute instructions.
05:49:37 Q. All computer systems are deterministic?

05:49:40 A. All the computer systems we're talking about here. There are some that are not deterministic.
05:49:44 Q. And those would be things like artificial intelligence?
05:49:48 A. No, something else.
05:49:50 Q. On that score, you don't think these patents claim some sort of thinking computer, do you?
05:49:55 A. No, they just claim a computer that creates paths dynamically.
05:50:00 Q. You don't think these patents claim some sort of a computer that has the ability to think for itself, do you?
05:50:06 A. I do not.
05:50:06 Q. And all computers are programmed, correct?
05:50:09 A. Yes, they are.
05:50:10 Q. And do you think that any computer that's programmed cannot infringe these claims because it is programmed?
05:50:18 A. I do not believe that at all.
05:50:19 Q. That would be a ridiculous position, correct?
05:50:23 A. I wouldn't want to characterize it. I just don't believe it.
05:50:26 Q. Do you recall I took your deposition, sir?
05:50:28 A. Yes, I do recall.
05:50:29 Q. And do you recall I asked you that very question?
05:50:31 A. I know you asked a lot of questions like that, yes.
05:50:35 Q. Do you recall that one?
05:50:36 A. I'm sorry?

05:50:36 Q. Do you recall that one?

05:50:37 A. Honestly, not specifically, but I'm happy to believe 05:50:42 that you asked me exactly that question.

05:50:44 MR. HOSIE: If we may have Page 59 of Dr. Jeffay's 05:50:48 testimony up. This is not the right one. I'll have to 05:51:01 find it. I am tired.

05:51:01 Your Honor, this would be a good time for a break. 05:51:01 I'm about to move into some other material, if it suits the 05:51:03 Court's convenience.

05:51:03 THE COURT: Approach the bench, counsel.

05:51:05 (Bench conference.)

05:51:14 THE COURT: Are you trying to tell me that you're 05:51:16 going to go back into confidential information? What are 05:51:19 you indicating?

05:51:20 MR. HOSIE: I'm just moving on to various other 05:51:21 subjects.

05:51:22 THE COURT: Well, I do agree this is probably as 05:51:27 good a time to break as any, especially given the length of 05:51:31 time this entire cross is going to take.

05:51:33 We'll recess for the evening, and we'll reconvene 05:51:38 in the morning.

05:51:40 I'm going to want to see you gentlemen in chambers 05:51:44 briefly after we recess.

05:51:46 MR. HOSIE: Thank you, Your Honor.

05:51:51 (Bench conference concluded.)

05:51:52 THE COURT: Ladies and gentlemen, this
05:51:54 cross-examination has some considerable length to go.
05:51:57 We're very close to 6:00 o'clock, and so I'm not going to
05:52:02 continue at this point any longer for today. We're going
05:52:05 to recess for the evening.

05:52:06 I'm going to ask you to be back tomorrow morning
05:52:10 prepared to go by 8:30, assembled in advance of that time,
05:52:14 as you were this morning.

05:52:15 I'll ask you to take your juror notebooks as you
05:52:18 leave the courtroom and leave them closed on the table in
05:52:20 the jury room.

05:52:22 I'll remind you follow all my instructions about
05:52:25 your conduct, including, of course, not to discuss the case
05:52:28 or anything about it with anyone. Please travel safely to
05:52:31 your homes. And we will see you tomorrow morning at or
05:52:35 about 8:30.

05:52:36 With that, the jury's excused for the evening.

05:52:39 COURT SECURITY OFFICER: All rise.

05:53:10 (Jury out.)

05:53:10 THE COURT: All right. Please be seated.

05:53:11 Counsel, for your purposes, let me update you. At
05:53:17 this point, the Plaintiff has remaining a total of 2 hours
05:53:21 and 31 minutes.

05:53:22 And the Defendant has remaining a total of 3 hours
05:53:26 and 3 minutes.

05:53:27 I'd like to see lead and local counsel in chambers
05:53:33 after we recess.

05:53:35 Be prepared in the morning before I bring the jury
05:53:37 in to read into the record the items from the list of
05:53:42 pre-admitted exhibits used during today's portion of the
05:53:46 trial.

05:53:47 Continue your commendable meet-and-confer efforts
05:53:49 overnight. The last few days you've not brought me any
05:53:54 disputes in the morning, and I commend you for that.
05:53:57 Continue that process this evening, as well, hopefully to
05:54:00 the same result. But, if not, I'll be available not later
05:54:02 than 7:30 in the morning in chambers.

05:54:06 With that, is there anything further from either
05:54:08 side that needs to be taken up before we recess for the
05:54:11 evening?

05:54:12 Anything further from Plaintiff?

05:54:14 MR. DAVIS: No, Your Honor.

05:54:15 THE COURT: From Defendant?

05:54:16 MR. GILLAM: No, Your Honor.

05:54:16 THE COURT: All right. We stand in recess until
05:54:24 tomorrow morning.

05:54:24 COURT SECURITY OFFICER: All rise.

(Recess.)

CERTIFICATION

I HEREBY CERTIFY that the foregoing is a true and correct transcript from the stenographic notes of the proceedings in the above-entitled matter to the best of my ability.

/S/ Shelly Holmes 12/11/19
SHELLY HOLMES, CSR, TCRR Date
OFFICIAL REPORTER
State of Texas No.: 7804
Expiration Date: 12/31/20